Personal Computer

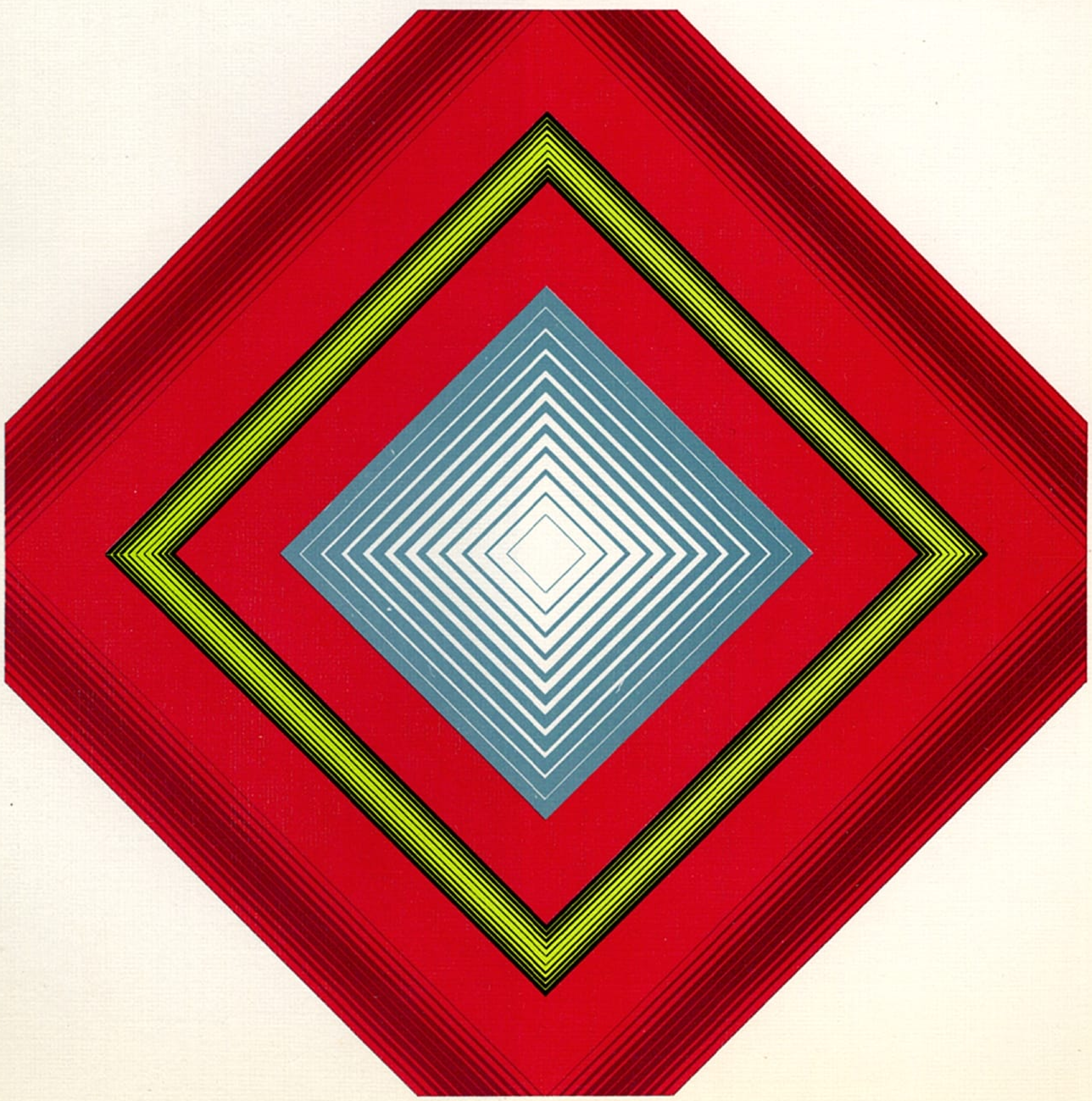
PC-8801 mkII

NEC

グラフィックス入門

脇英世＝著

ナツメ社



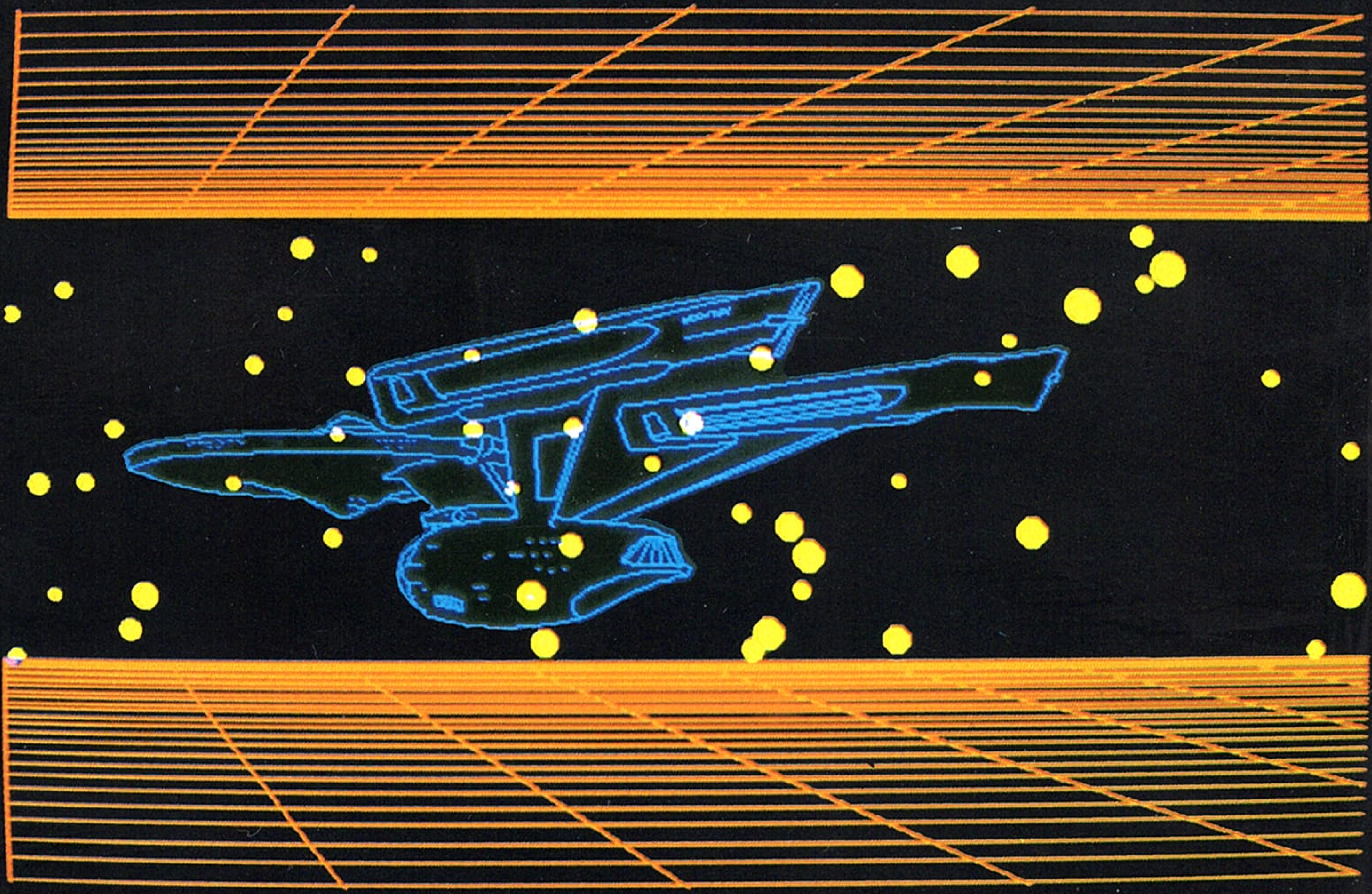
Personal Computer
PC-8801mkII
NEC
グラフィックス入門
脇英世＝著
ナツメ社

[本書を読まれる方へ]

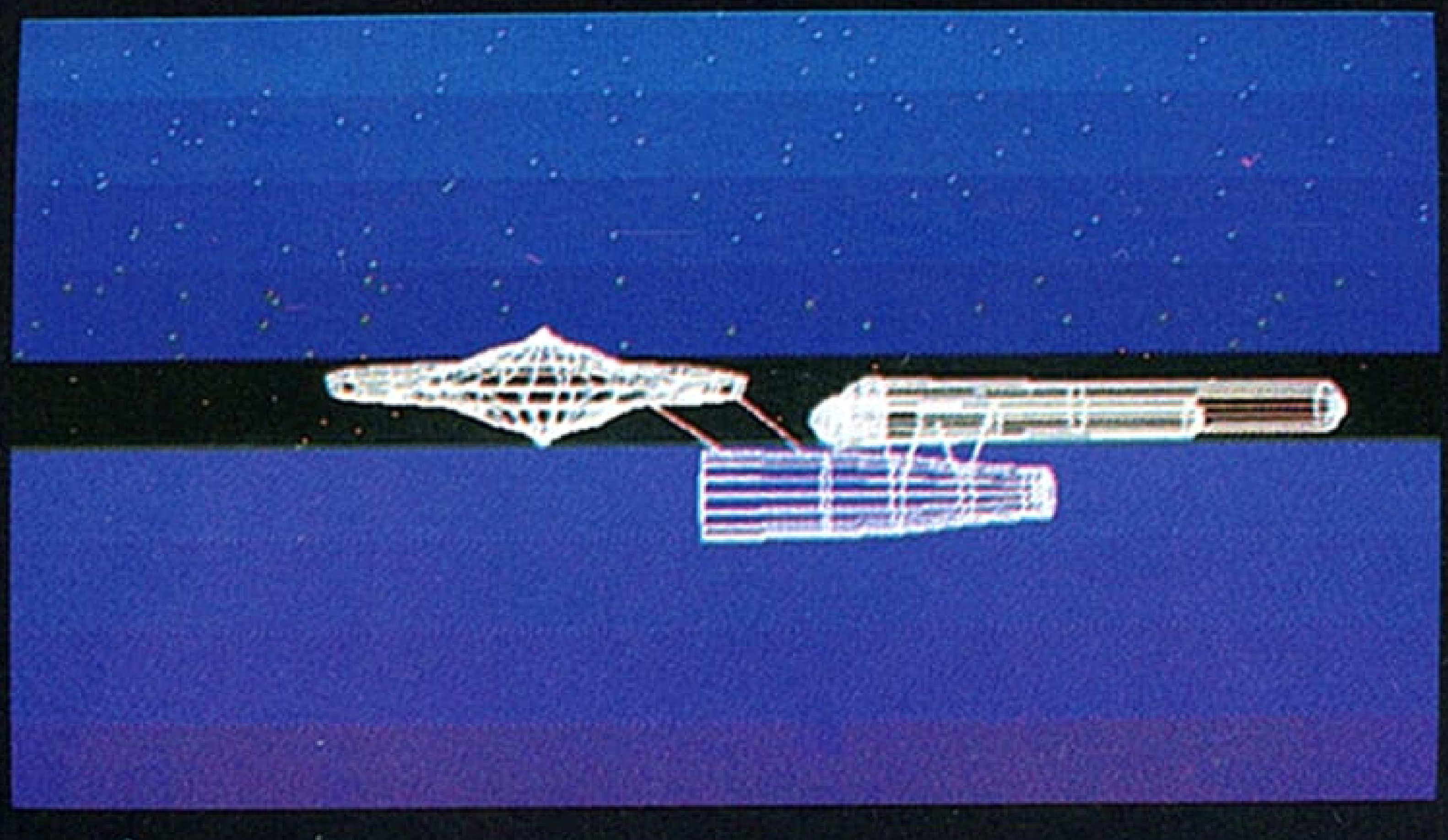
⊙本書は、スーパーパソコンPC-8801の後続機
PC-8801mkⅡを使ったグラフィックス入門書です。
⊙グラフィックスはビジネス、ゲームプログラミングと比べて
その考え方を異にするためむずかしいといわれています。
とくに広大な空間をもつワールド座標系、
あるいはスクリーン座標系、
ビューポートなどが導入されるため、
よりいっそう入りにくくしているようです。
⊙しかし、これはイメージ、感じ方の違いで、
慣れてしまえばそれほどでもありません。
⊙本書では、入門編を終えた方が
抵抗なく進めるように基本的コマンドから解説し、
またグラフィック座標系をイメージ的に把握できるよう
巻末に補足説明を付しました。
⊙まず、色の塗り方、簡単な図形の描き方の
短いサンプルプログラムを実行して、
グラフィックスの考え方をあなたの眼で確かめてください。
このようにして少しずつステップを進めるごとに、
あなたのグラフィックスへの関心は
ますます深まることでしょう。
⊙デザインに、ビジネスに、そして趣味として、
グラフィックスには限りない魅力が秘められているのです。

cover design —— nobuo nakagaki + yoshifumi hayase

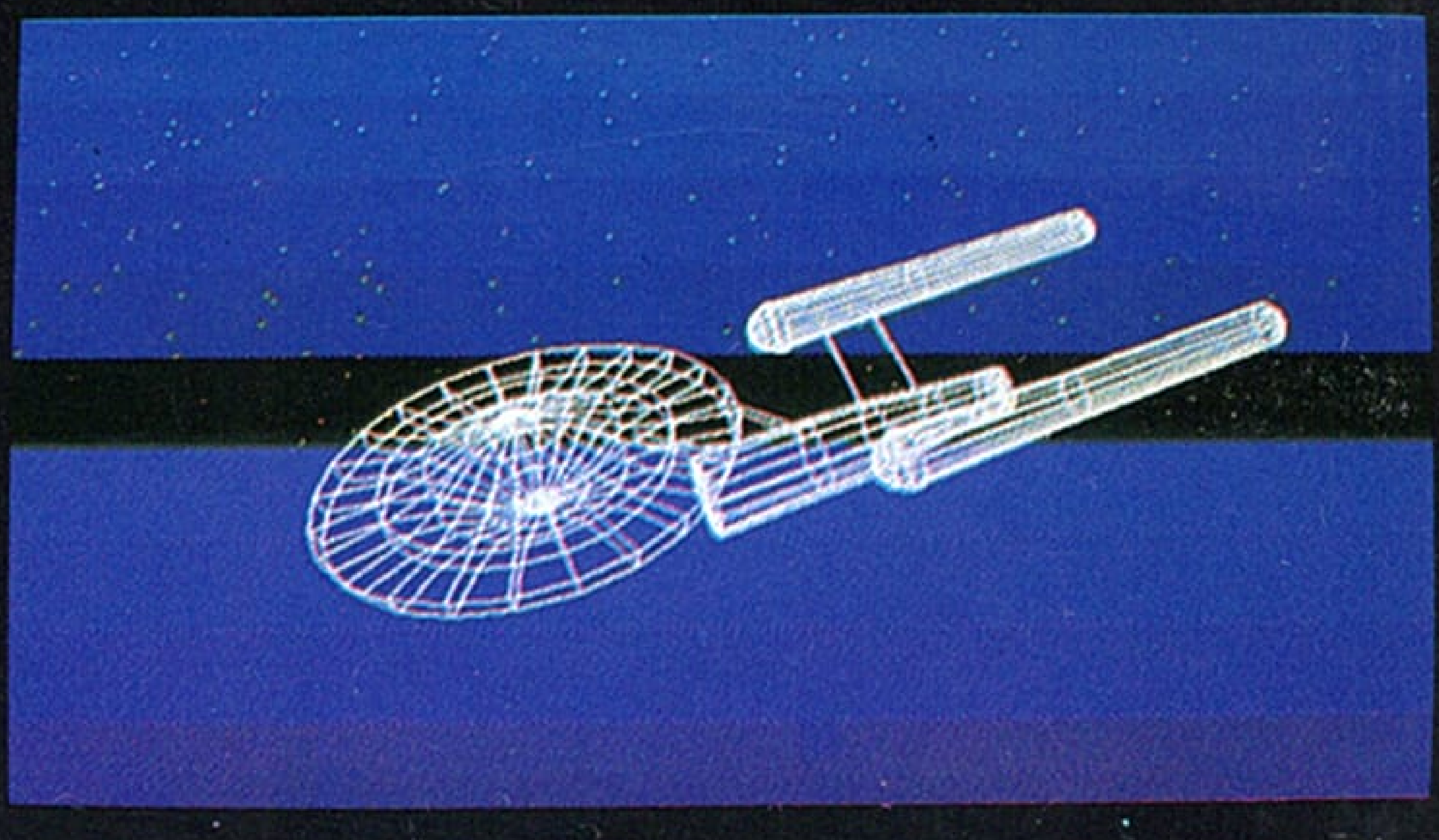
PC-8801mkII GRAPHICS



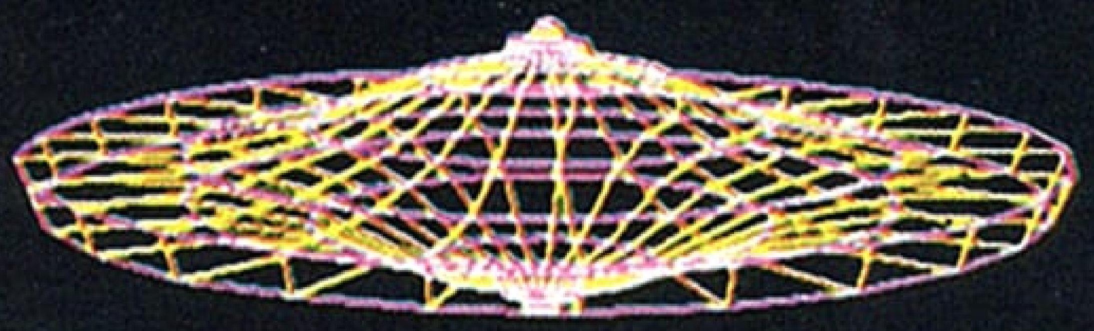
1



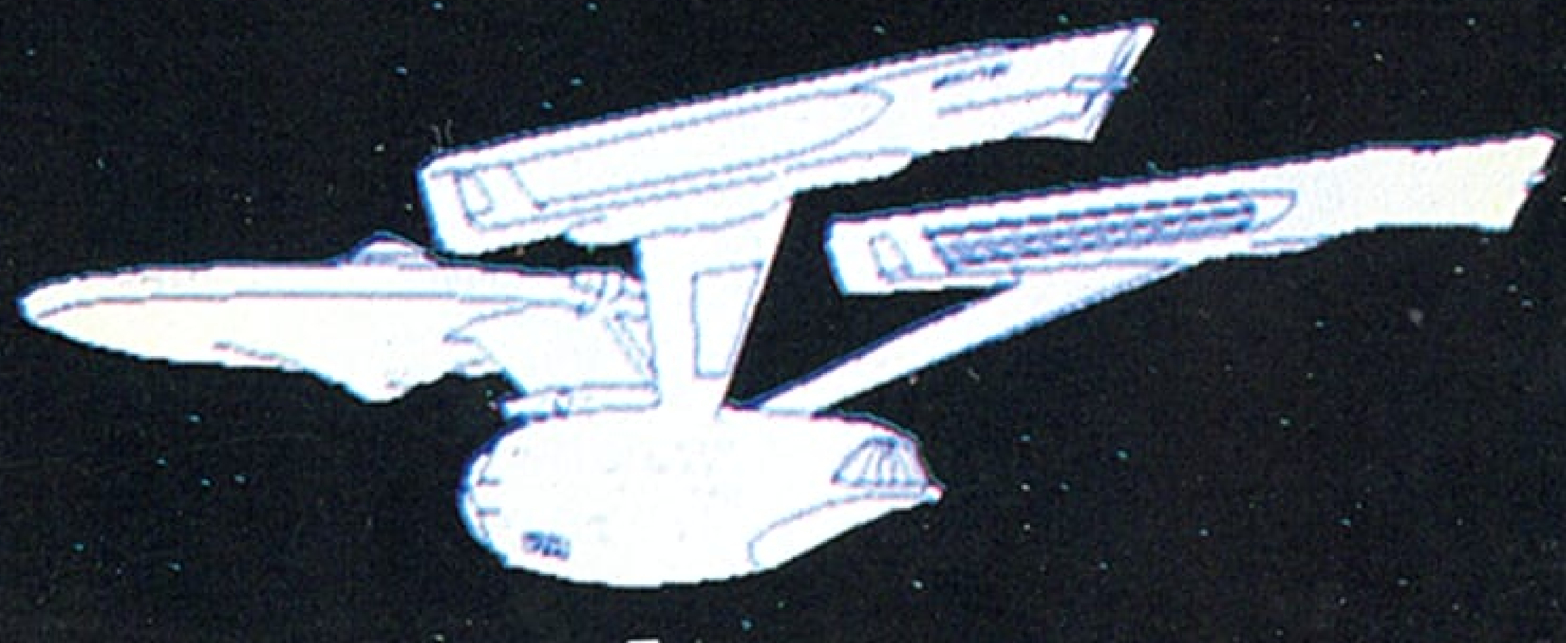
2



3

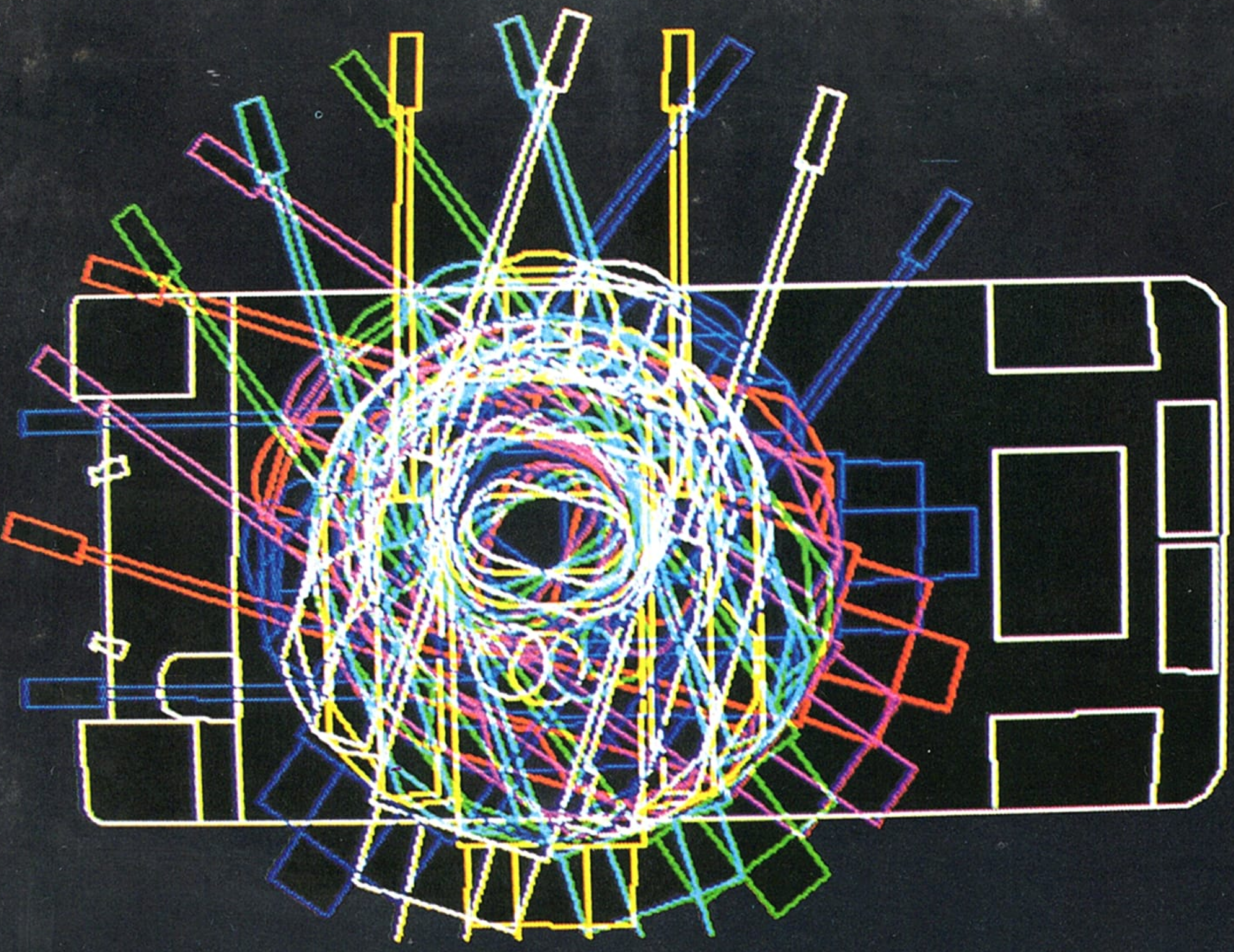


4

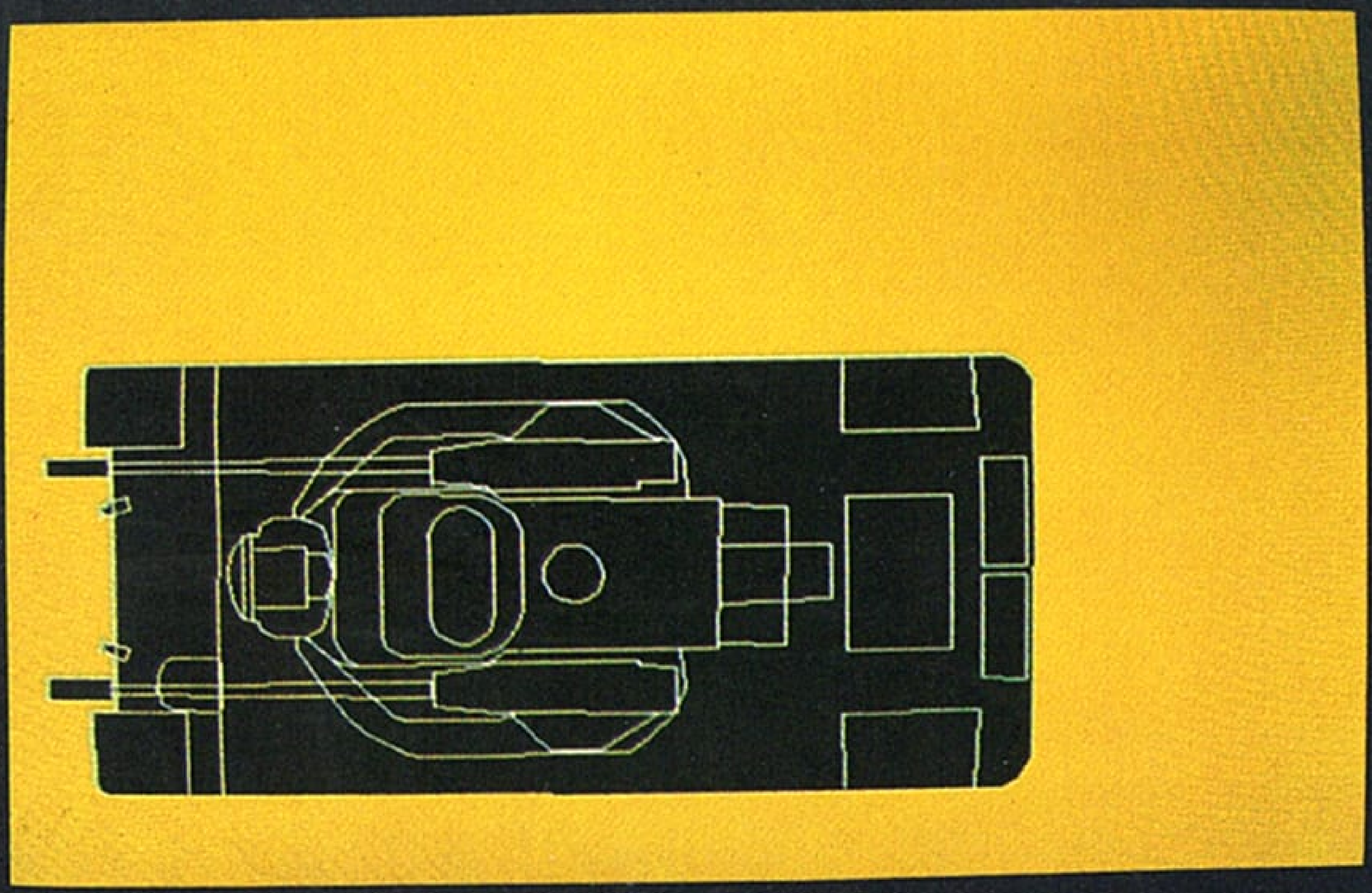


5

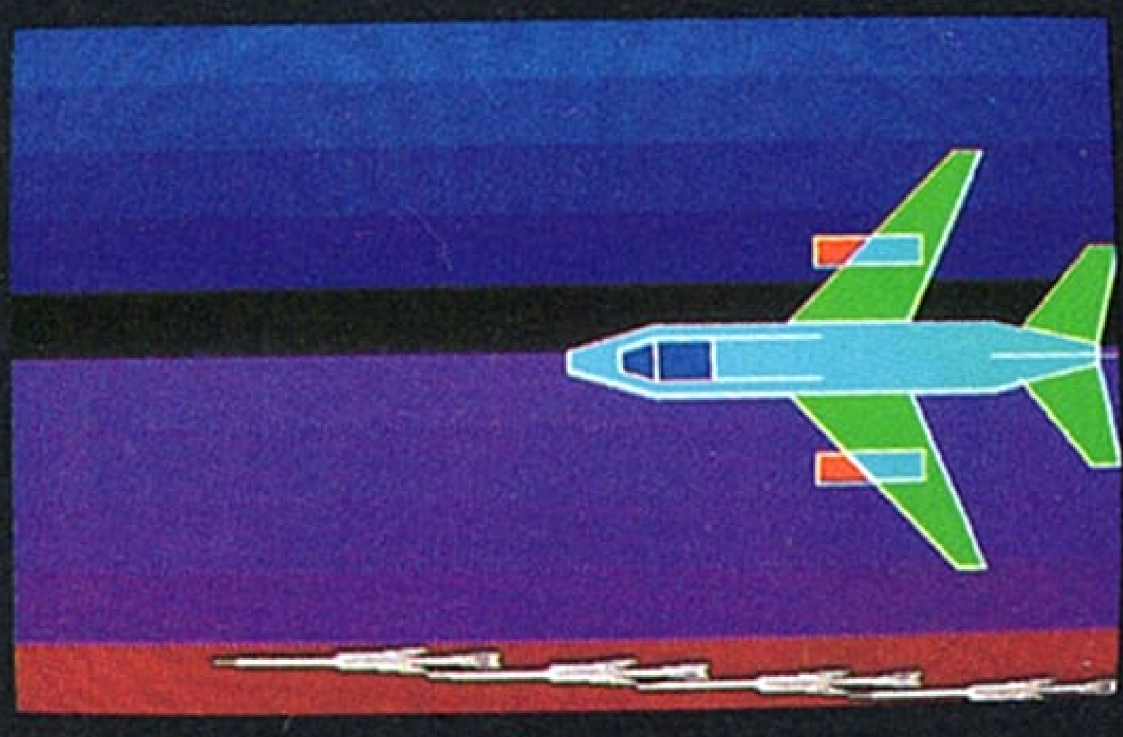
1── スターゲートと宇宙船→P.65
2── 宇宙を航海するエンタープライズ号→P.183
3── 降下態勢のエンタープライズ号→P.183
4── エンタープライズ号の艦橋部分→P.174
5── 暗黒の宇宙を漂うエンタープライズ号→P.60
⊙この口絵画面は本文中のプログラムに
多少手を加えカラフルにしました.



6

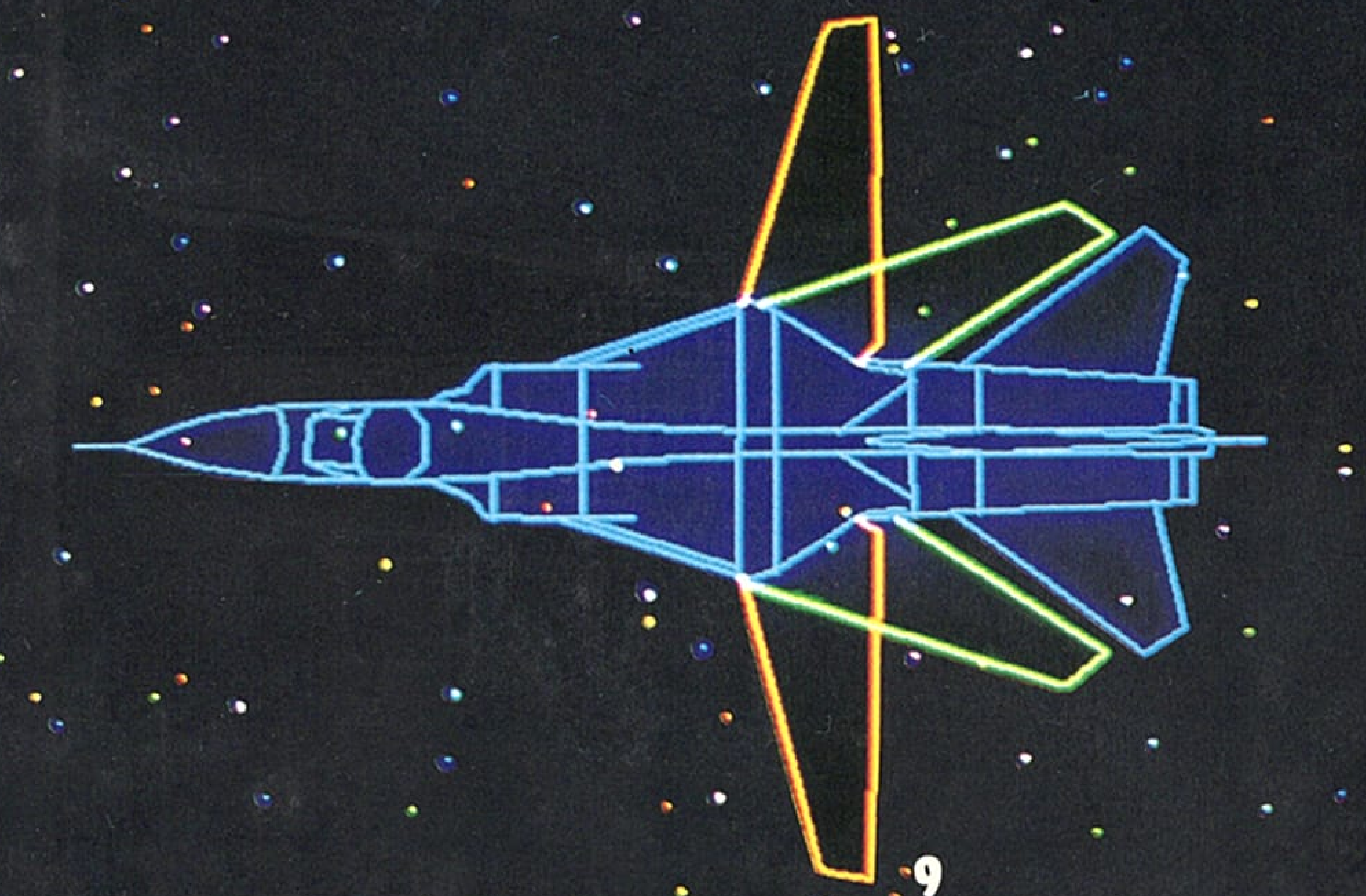


7



8

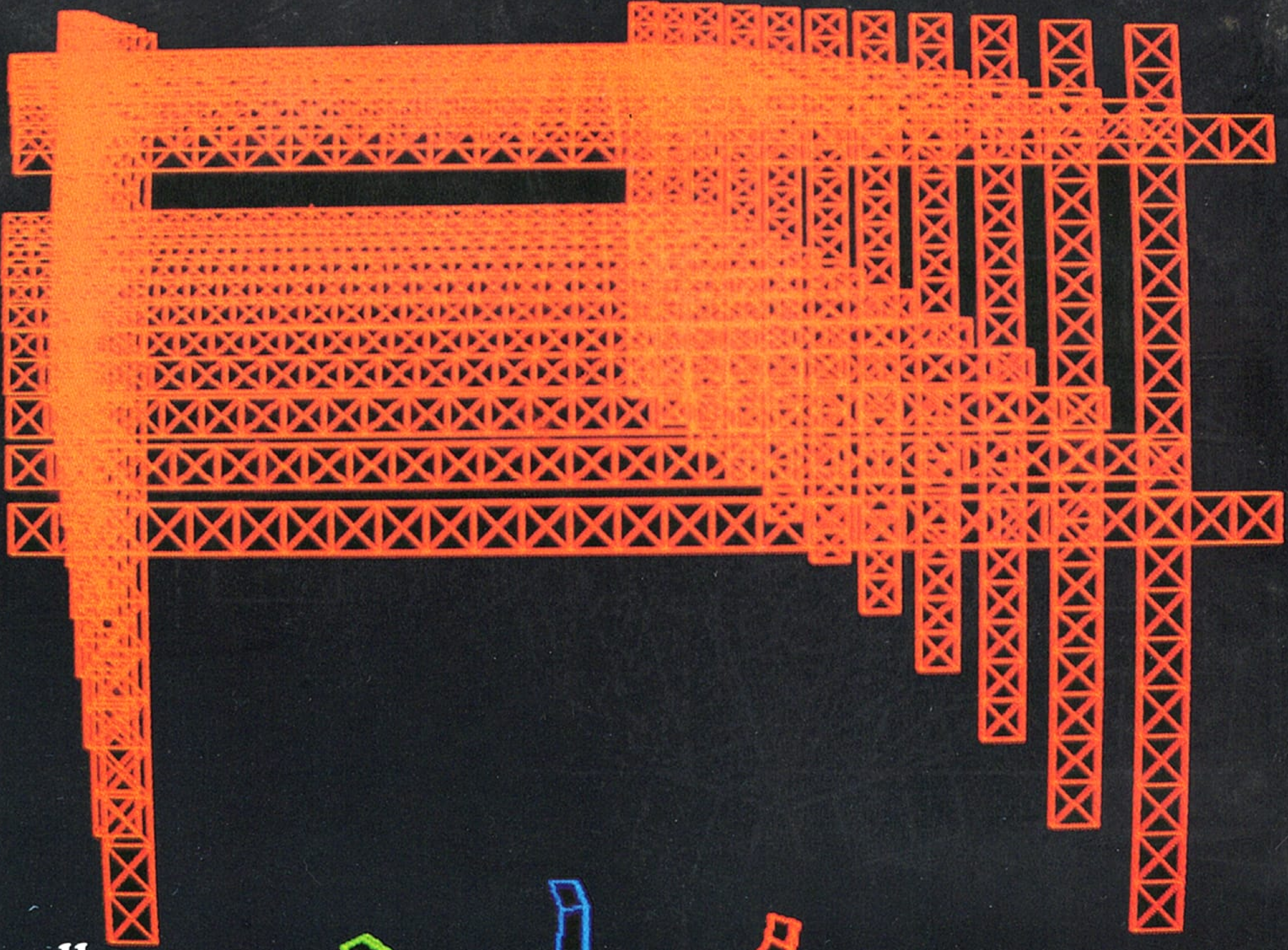
6——上物構造物旋回中の図→P.145
7——車輛上面図→P.125
8——反転ひき起しに入ったジェット戦闘機→P.77
9——可変後退翼ジェット機→P.77
10——ジェット機の編隊→P.94



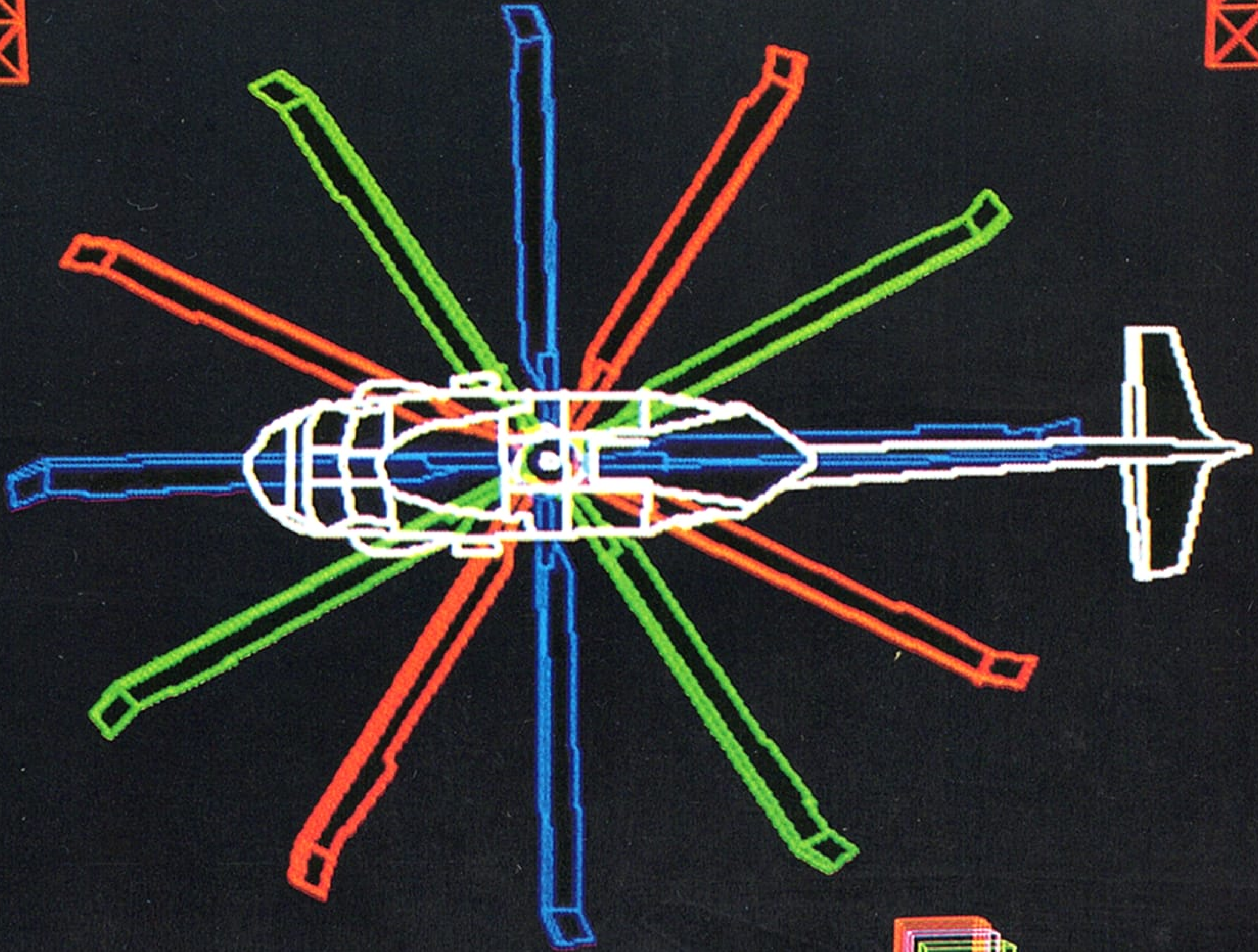
9



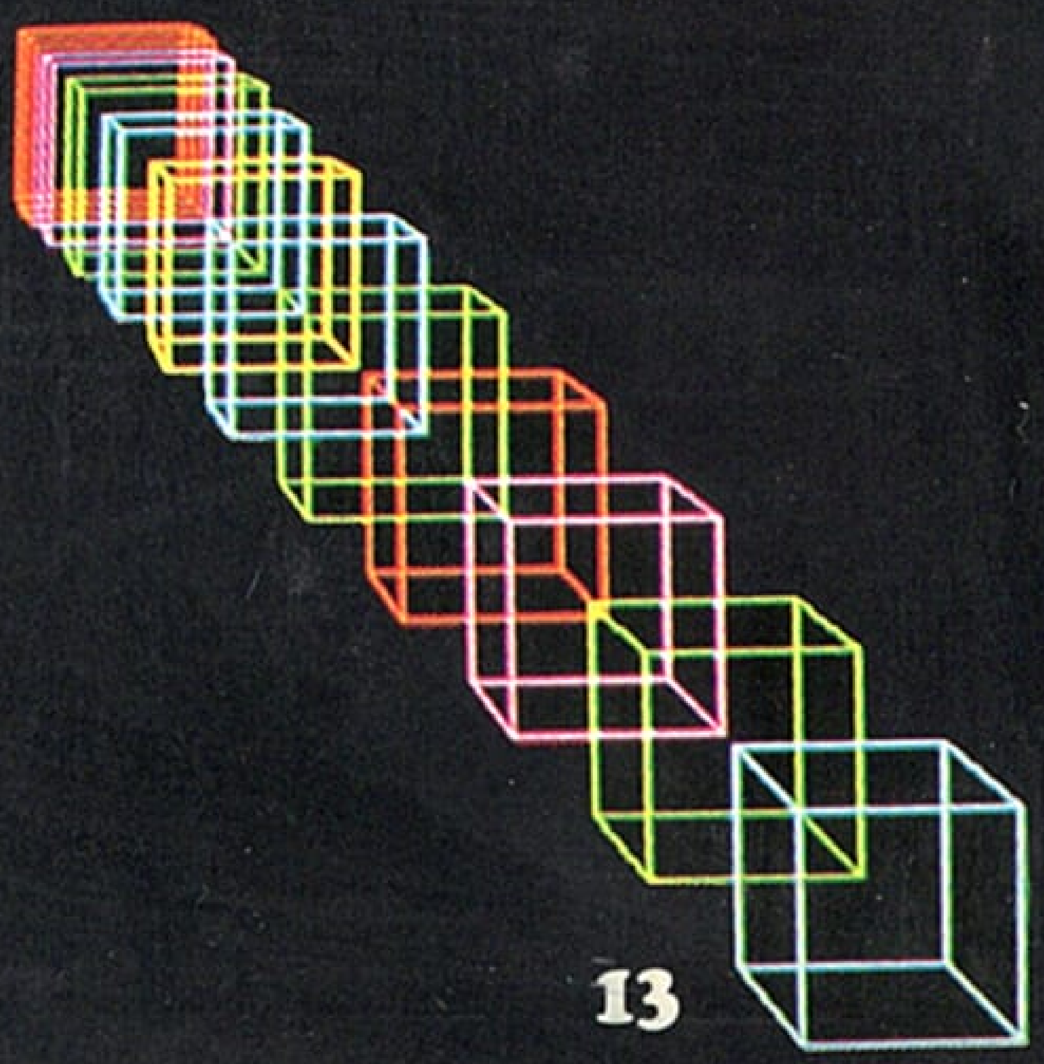
10



11



12

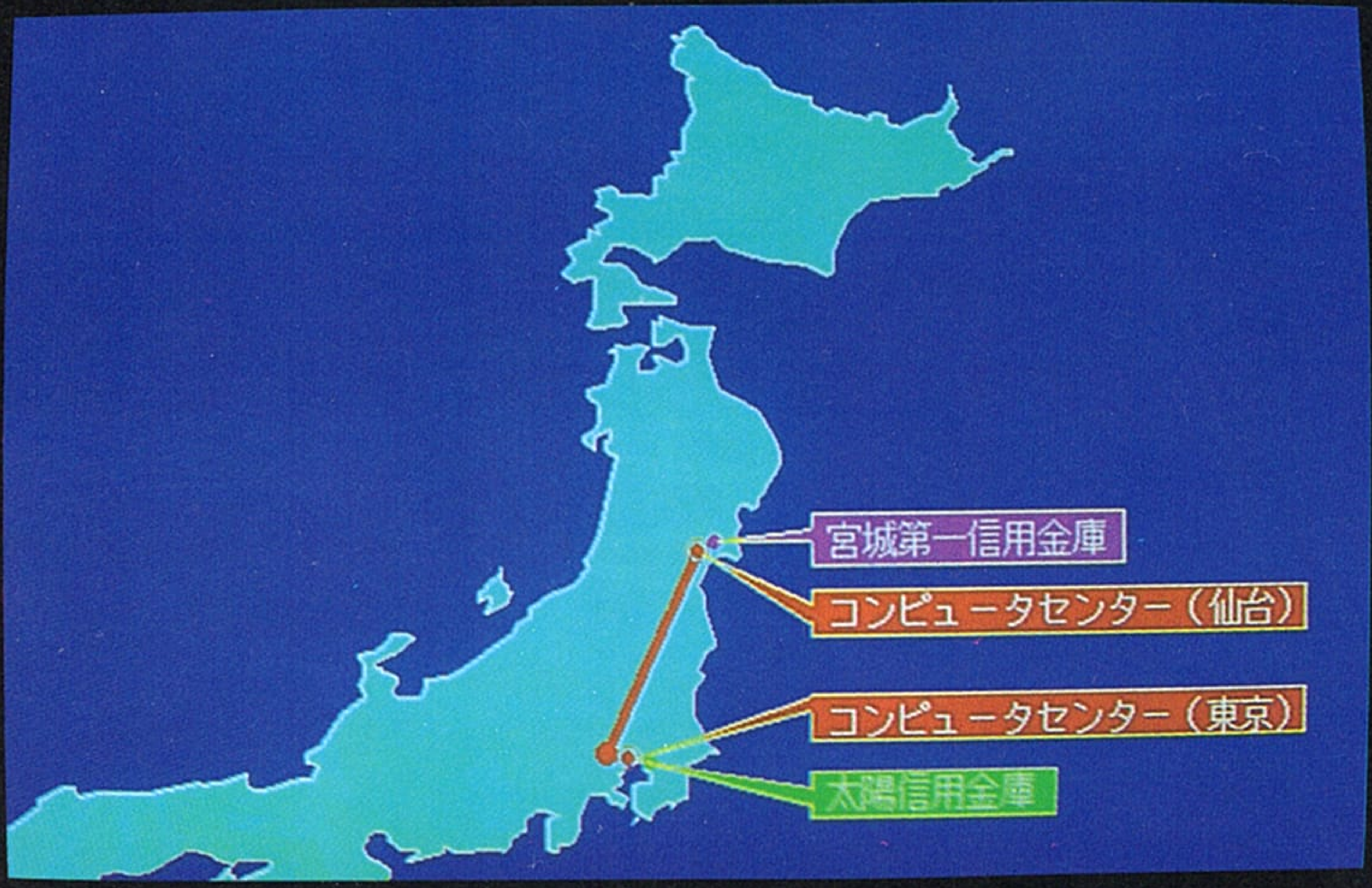


13

11――枠組で作った構造物→P.106
12――ヘリコプターの回転翼→P.71
13――立方体を平行移動させる→P.155
⊙この口絵画面は本文中のプログラムに
多少手を加えカラフルにしました。

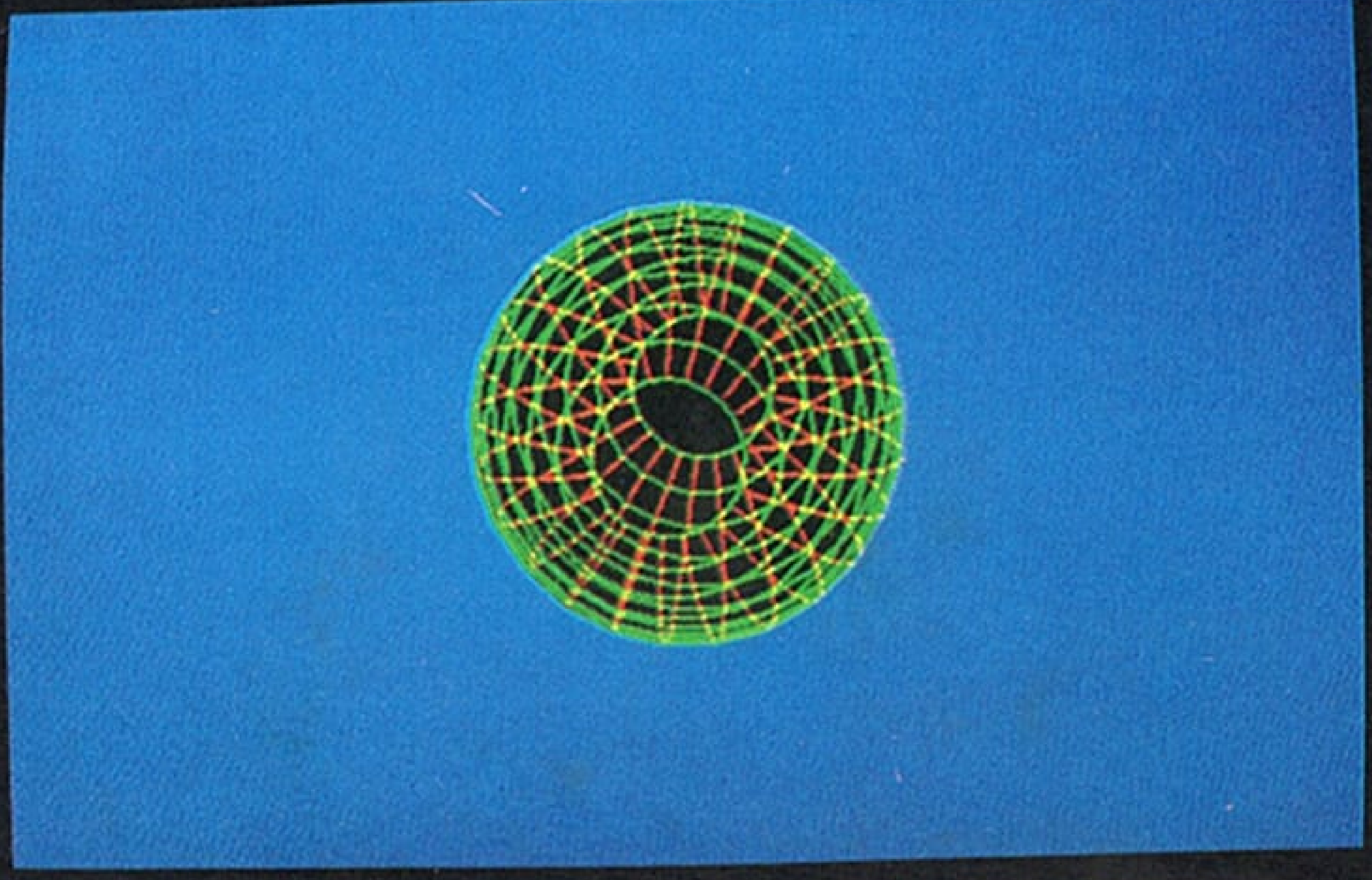


14



15

16



14 ── TV用オンラインシステムの図
漢字使用が有効である➡P.86
15 ── TV用の日本地図画面➡P.86
16 ── 穴の開いた地球➡P.173

⊙14、15の画面プログラムは掲載されていません。

Personal Computer

PC-8801

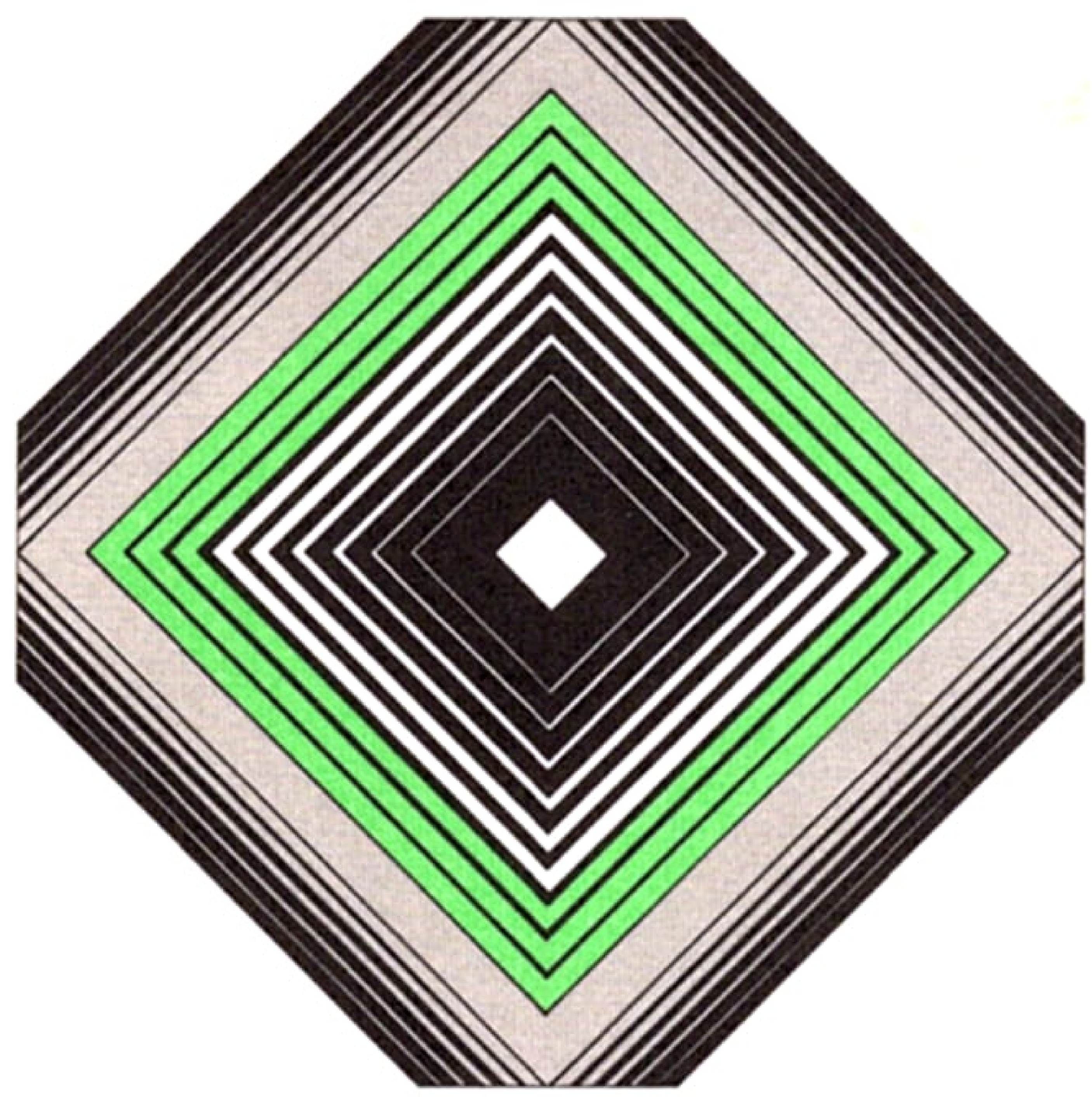
NEC

mkII

グラフィックス入門

脇英世＝著

ナツメ社



はじめに

パソコンでグラフィックスをやってみることは長い間の私の夢でした。しかし、パソコンの能力が限定されている間はどう歯ぎしりしても、かなわぬ夢であったと言えます。テクトロニクスやヒューレット・パッカード、DECなどの展示会ですばらしいコンピュータ・グラフィックスを見るたびにいつか私もパソコンで、こういうすばらしいものを楽しんでみたいと思っていました。

ところが驚異的なパソコンの技術革新の結果、その日は意外にはやくやってきました。私の予想ではあと数年先でないと無理だと思っていたのですが、その日は予想以上に早くやってきました。

マイクロ・エレクトロニクス技術の結晶であるスーパー・パソコンＰＣ-８８０１は私の夢をかなりの程度までかなえてくれました。8ビットCPUのZ-80でありながら、バンク切りかえの手法によってメモリ空間を大きく拡張し、グラフィックをやりやすくしています。新しい斬新なグラフィック命令もたくさん盛りこまれました。WINDOWやVIEWなどが3次元グラフィックスをたいへんやりやすくしていますし、パレットやPAINT TILE文は色彩の世界を大きく広げてくれました。その他盛りだくさんのグラフィック機能が準備されています。

さらに私の力を何倍にもしてくれたのは、研究室の若い学生諸君の存在です。ここで展開しているグラフィックスを作り上げるのは私個人の力ではとても不可能でした。十数人の若い学生諸君の応援があって、始めてこのようなことができたのです。精神的にも体力的にも、この仕事は大変負担の重い仕事でした。そして新しい斬新なアイデアがいくつも若い学生諸君の柔軟な頭から生みだされました。

当初の計画では目次案というものは全くなく、私が好きなようにやりたいことを思いついては原稿を作っていったので、編集を担当されたナツメ社の田村正隆氏の御苦労は大変だっただろうと推察されます。私の説明不足の点は田村さんが、かなり補筆して下さいました。

子供の頃、絵を教えて下さった先生が、『君は花鳥風月と人間を全く描かない変った子供だね』。とおっしゃられましたが、20年以上ぶりに絵らしいものを描いても、あまり変っていませんでした。いつかは、『オズの魔法使い』とか『ピノキオ』の世界に挑戦しようとは思っていますが、今回は私の好きなようにやってしまいました。深く反省し、次回からはソフトな題材にするようにしたいと思います。

1984年１月　　脇　英世

編集部より読者の方へ

1983年12月にＮＥＣよりＰＣ―8801ＭＫⅡが発売になりました。従来のＰＣ―8801に比べ、ずいぶんコンパクトにまとめられ、大変使いやすくなっています。

搭載されているBASICはほとんど変わりませんが、拡張機能がソフトにより追加され、サウンド機能、タートルグラフィック機能などを楽しむことができるようになりました。

また、ハード面でのすばらしい前進は、モデル10は別として、モデル20およびモデル30にディスクユニットが内蔵されたことです。カセットテープ時代は、セーブ、ロードはともかく、データの扱いとなるとお手あげの状態でした。

ことにグラフィックスにおいては、気に入った描画を楽しもうと思えば、どうしてもデータ処理を余儀なくされますし、その量も必然的にふえていきます。そこで威力を発揮するのがディスクです。あるデータをディスクにたくわえておけば、プログラムを変えることにより、そのデータを違った形で使うことができるのです。

本書は、ディスクの内蔵されていないモデル10の方にも使えるように執筆されていますが、編集部ではモデル20、30をお持ちの方を対象に、ＤＡＴＡ文からどのようにデータファイルを作り、それをどのようにＢＡＳＩＣで呼び出すかを補足説明として巻末に付しました。

また、グラフィックスに入ってはじめにつまづくのが座標系の考え方です。進んだ読者が大多数と思いますが、なかには、もう少し説明があったらと思う方もいると思いますので、座標系のイメージをつかんでいただくための解説を巻末に補足しました。

なお、本文中の記事に対して補足説明があるものは、その該当箇所に次のような表示を付しました。

☞ **1 (P.203)**

この場合、この表示のある付近の補足説明は203ページにあります。

なお、本文中の機種名はＰＣ―8801に統一してありますのでご諒承ください。

ご意見、ご希望がございましたら、「ナツメ社パソコン編集部」宛てお寄せくださるようお願いいたします。

編集部

もくじ

第0章

グラフィック画面について



0・0　始める前に

コンピュータ・グラフィックスというとすぐ思い出すのは『スターウォーズ』や『TRON』のような映画のすばらしい特撮シーンです。少し古い話ですけれども、学生時代にスタンリー・キューブリック監督の『2001年宇宙への旅』を初めて見た時の感動は忘れることができません。アーサー・クラークというＳＦ作家の原作をその後読みましたが、説明的にすぎて映像の世界から受けたほどの感動はなかったように覚えています。今でも『2001年宇宙への旅』のVTRを見ていますと、1968年という昔に良くこんなにすばらしいコンピュータ・グラフィックスが盛り込めたと感心するばかりです。

さきに私は『PC-９８０１グラフィックス入門』という本を書きました。この本の内容は必要な修正を加えれば、ほぼPC-８８０１にも適用できるものですが、初心者のかたにはその必要な修正が何であるかがわかりにくいのではないかという御指摘を頂きましたので稿をあらため、PC-８８０１用に書き直すことにしました。プログラムの一部を修正するだけでも良いと思ったのですが、こういうチャンスはなかなか来ないものですし、一度出版するとなかなか手直しできないものです。そこで手持ちの材料を使って、PC-９８０１の本の時に不本意に感じていた部分や、もうすこし面白い例題があるのではないかと考えた部分を直すことにしました。流れに大きな変更を加えずに手直しをすることは案外面倒な事柄でしたが、春や夏の休暇中に一生懸命アルバイトをしてやっとの思いでＰＣ-８８０１を買った学生諸君のことを考えると励みになりました。

本書はPC-８８０１のビジネス用アプリケーションとか、BASICの解説とかは対象にしていません。それはPC-８８０１にはすぐれた厚いマニュアルが３冊もついてきますので、これを読めば十分だからです。そこで本書ではPC-８８０１を使ったコンピュータ・グラフィックスだけを取り扱いました。ただし対象とするのは高級技術者やセミプロ級の学生ではなく、ごく一般の方々を想定しています。グラフィックスに必要はBASICはこの本だけでわかるようにしてあります。厚いマニュアルのどこから読んだら良いかわからない方々にとってはBASICの入門書ともなると思います。PC-８８０１のBASICの文法は大変高級でこみいった所もあるので、構造図をつけました。

0・1　準備するもの

これからコンピュータ・グラフィックスを勉強して行く上で必要なものをあげておきましょう。

❶　ＰＣ―8801
モデル10、モデル20、モデル30、いずれでもかまいません。

❷　ＣＲＴディスプレイ
640×200ドットのカラー型または640×400ドットのカラー型

❸　プリンタ
できればあると便利ですが、なくともかまいません。

❹　カセットテープレコーダ
これはモデル10の方のみです。

モデル20、モデル30にはディスクユニットが内蔵されています。ディスクがあるとプログラムのセーブ、ロードなど大変便利ですが、何よりも大きなメリットは、グラフィックデータをデータファイルとして保存しておけることでしょう。

本書は最小システムで学べるように執筆しましたので、ディスクでのデータファイルの扱いは巻末に掲載してあります。〈注〉

ＣＲＴディスプレイはカラーにしてください。白黒ではグラフィックスの良さを味わうことができません。予算に応じていろいろな型が用意されていますので、選ぶ楽しみも味わってください。

できれば640×400ドットのほうが良いのですが、640×200ドットでもかまいません。プログラムはどちらでも動かせます。

〈注〉ディスク操作およびディスクベーシックのコマンド等については解説してありませんので、初めてデータファイルを扱うという方は次の書籍を参照してください。

『ＰＣ―8801mkII　データファイル入門(Ｉ)』(ナツメ社)

0・2 テキスト画面の表示を決めること

FM-8を使いなれた私にとって、PC-8801やPC-9801は少しとまどうことがあります。それはPC-8801とPC-9801では文字の表示と画面の表示は別になっているということです。慣れてしまえば文字とグラフィックが別になっていることは大変便利です。なぜなら、せっかく画面にこしらえた表示がカーソルが画面にでただけでこわれることがないからです。

ディップスイッチ

文字の表示の仕方（テキスト画面の表示の仕方）とグラフィックの表示の仕方（グラフィック画面の表示の仕方）はPC-8801の**前面**にあるディップスイッチ（デュアル・イン・ライン・パッケージ・スイッチ）を切り換えるだけでもできますが、初めての方は間違えやすいので、スイッチでの切り換えでなく、ソフトウェアでの切り換えを覚えておいた方が良いでしょう。

フロッピーディスクをつながないでPC-8801の電源スイッチを入れた場合、画面には次のようなメッセージがでます。

```
How many files (0-15) ? ■
```

ここでは5ぐらいを入力して下さい。リターンキー ⏎ を押す流儀もありますが、クセになってしまうと困ることがあるのです。特に入出力装置（カセットテープレコーダ、フロッピーディスク装置、コミュニケーションポート、キーボード、画面、プリンタ）を使う上で制約がでてしまうので、必ず5ぐらいを入力して下さい。メモリーの一部が削られますが、PC-8801ぐらいになると大した問題ではありません。

5 ⏎ の状態から画面の表示を決定しておきます。

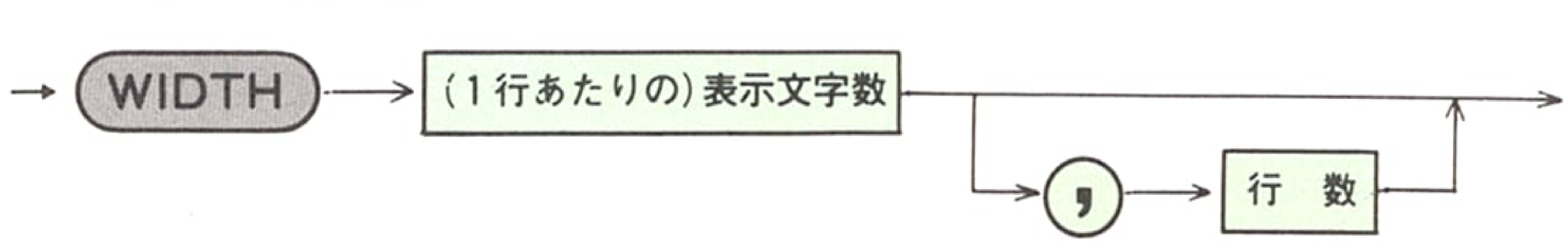
まず文字の表示を決めてやりましょう。文字の表示に必要な指定は

WIDTH ●width／ウィズ

目的 画面に表示する文字の桁数を指定する

書式 WIDTH 〈桁数〉〔,〈行数〉〕

文例 WIDTH 80, 25

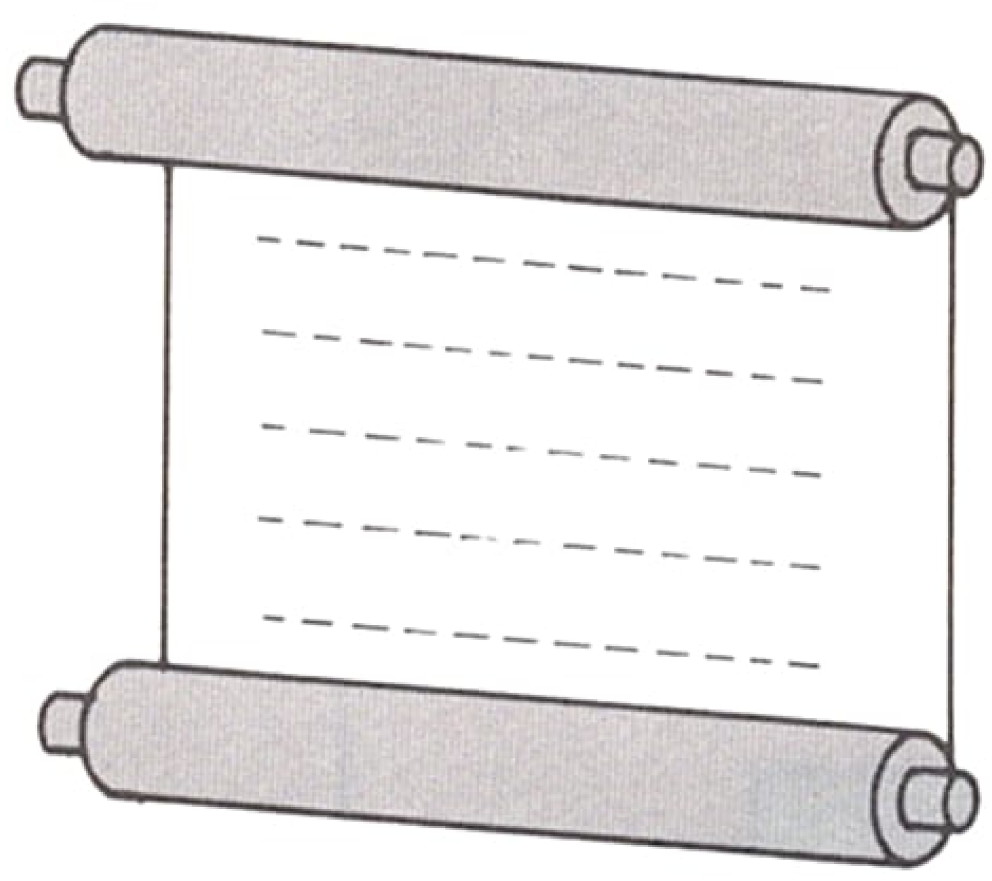


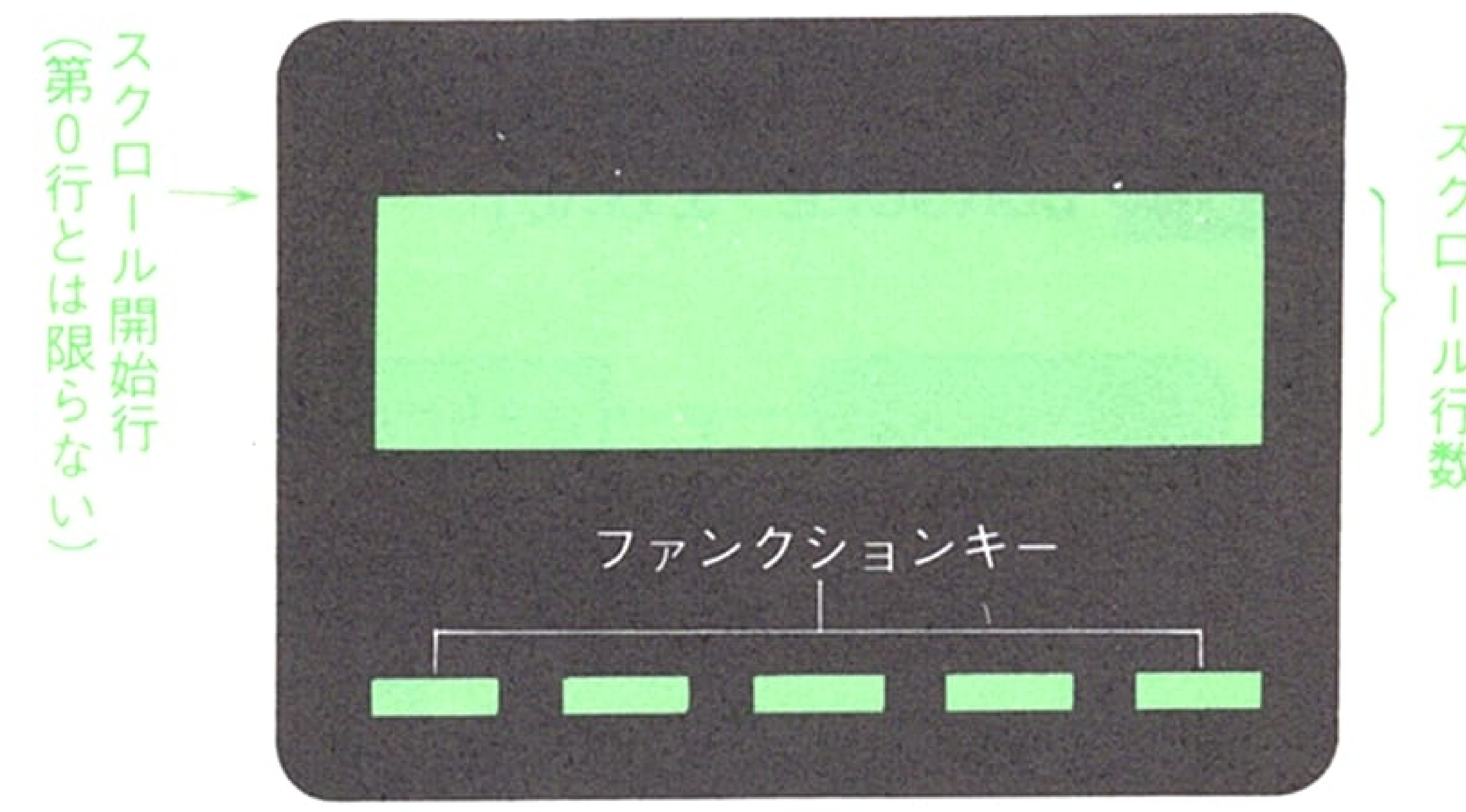
表示文字数とスクロール範囲の決定です。表示文字数は次の4通りが可能で、次のようにします。

横40桁×縦20行	WIDTH　40, 20
横80桁×縦20行	WIDTH　80, 20
横40桁×縦25行	WIDTH　40, 25
横80桁×縦25行	WIDTH　80, 25

どれか1つ選んで入力して下さい。

巻物のスクローリング





表示文字数を決定したらスクロール範囲とファンクションキー表示スイッチとカラー／白黒スイッチを指定することです。スクロールというのは巻物を巻きあげながら一部だけを見て行く操作で、CRT画面に何行表示するかを決めるものです。これらはCONSOLE（コンソール）文だけで決定されます。

たとえば、

ファンクションキーを表示しない
カラーモードの指定

```
CONSOLE 0,25,0,1
```

としますと、スクロール開始行は第0行で、スクロール行数は25で、ファンクションキー表示は出ず、テキスト画面はカラーモードになります。ファンクションキーが画面に出ると見にくいので、この指定にされると良いと思います。

案外簡単なことでエラーを出しやすいのが次の例です。

```
WIDTH 40,20
CONSOLE 0,25,0,1
```

これはプログラムを作るとき失敗しやすい例です。

それはWIDTH文で20行しか表示しないと宣言しているのに、CONSOLE文では25行表示しろと無理なことを命令しているからです。WIDTH文とCONSOLE文がこれだけ接近していると間違いがわかりやすいのですが、少し離れていたり、WIDTHの指定が電源投入時のままだと間違いになかなか気がつきにくいのです。

CONSOLE文のすべての並びを指定する必要はありません。

```
CONSOLE 0,25
```

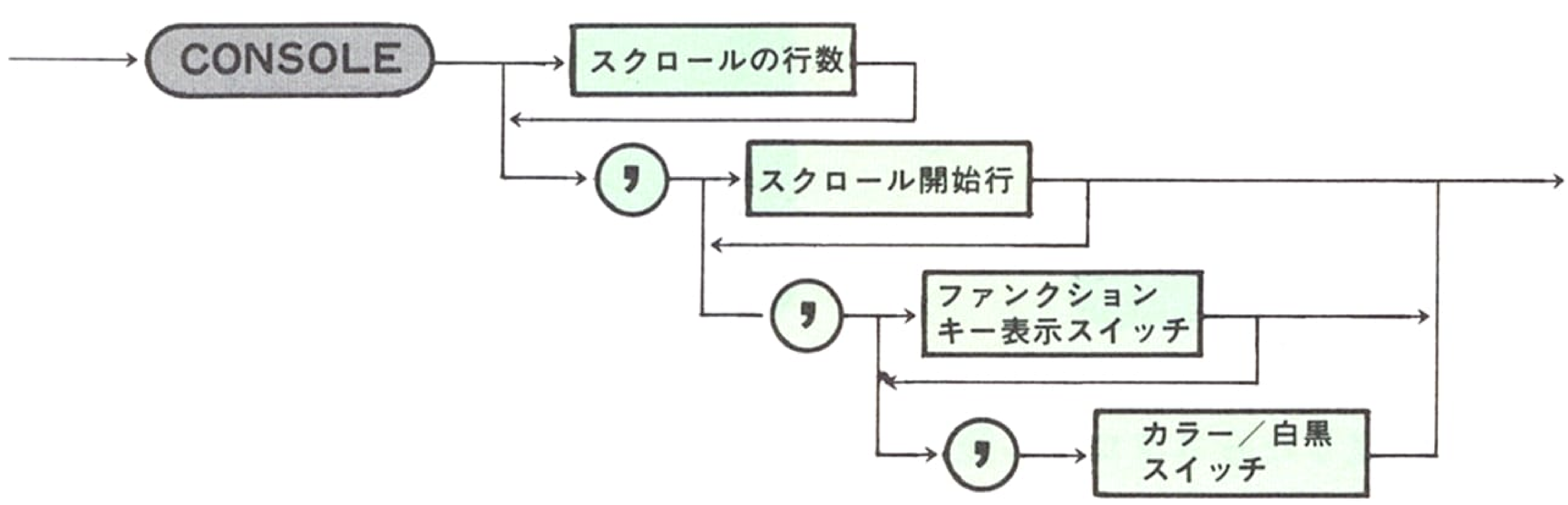
とすることも可能です。CONSOLE文の構造図を見て下さい。すべての並びを通らずに右側の出口に抜けることができますね。始めは難しそうですが、慣れて来ますと、これなしでは不便でやって行けなくなるくらい便利なものです。

CONSOLE ●console／コンソール

目的 テキスト画面のモードの設定を行う

書式 CONSOLE〔〈スクロール開始行〉〕,〔〈スクロール行数〉〕〔,〈ファンクションキー表示スイッチ〉〕〔,〈カラー/白黒スイッチ〉〕

文例 CONSOLE 0,25,0,1

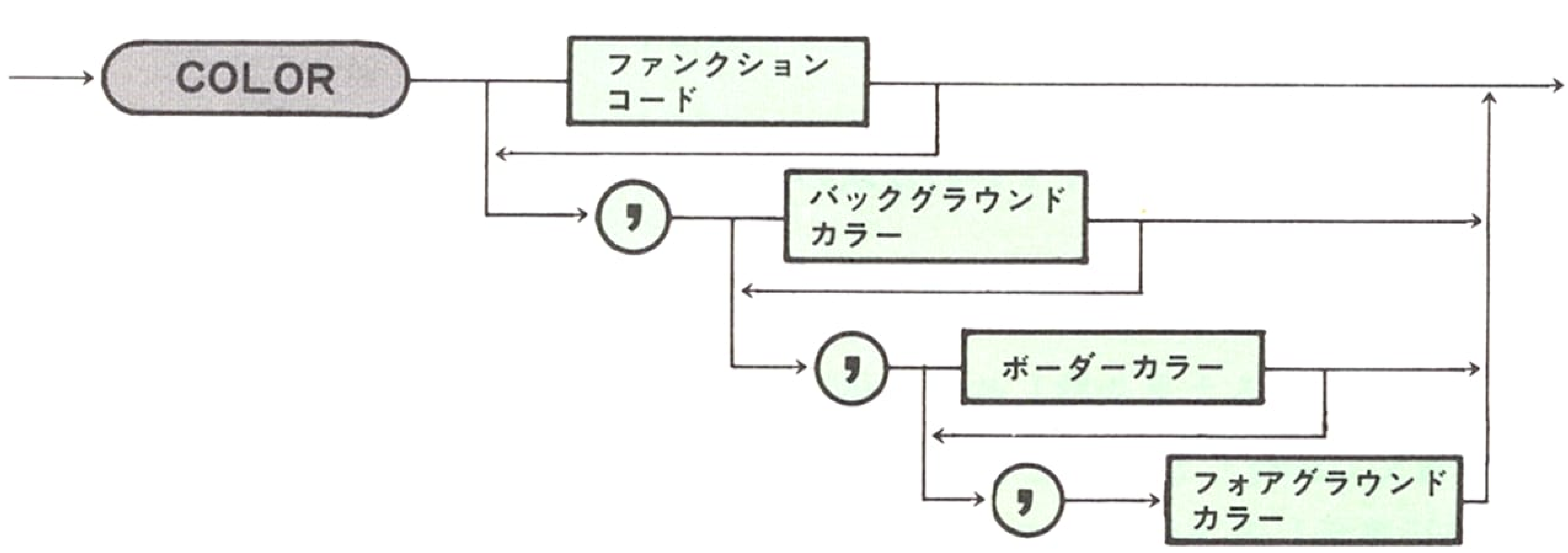


COLOR ●color／カラー

目的 ディスプレイ画面の各部の色を指定する

書式 COLOR〔〈ファンクションコード〉〕〔,〈バックグラウンドカラー〉〕〔,〈ボーダーカラー〉〕〔,〈フォアグラウンドカラー〉〕

文例 COLOR 7, 0, 0, 7



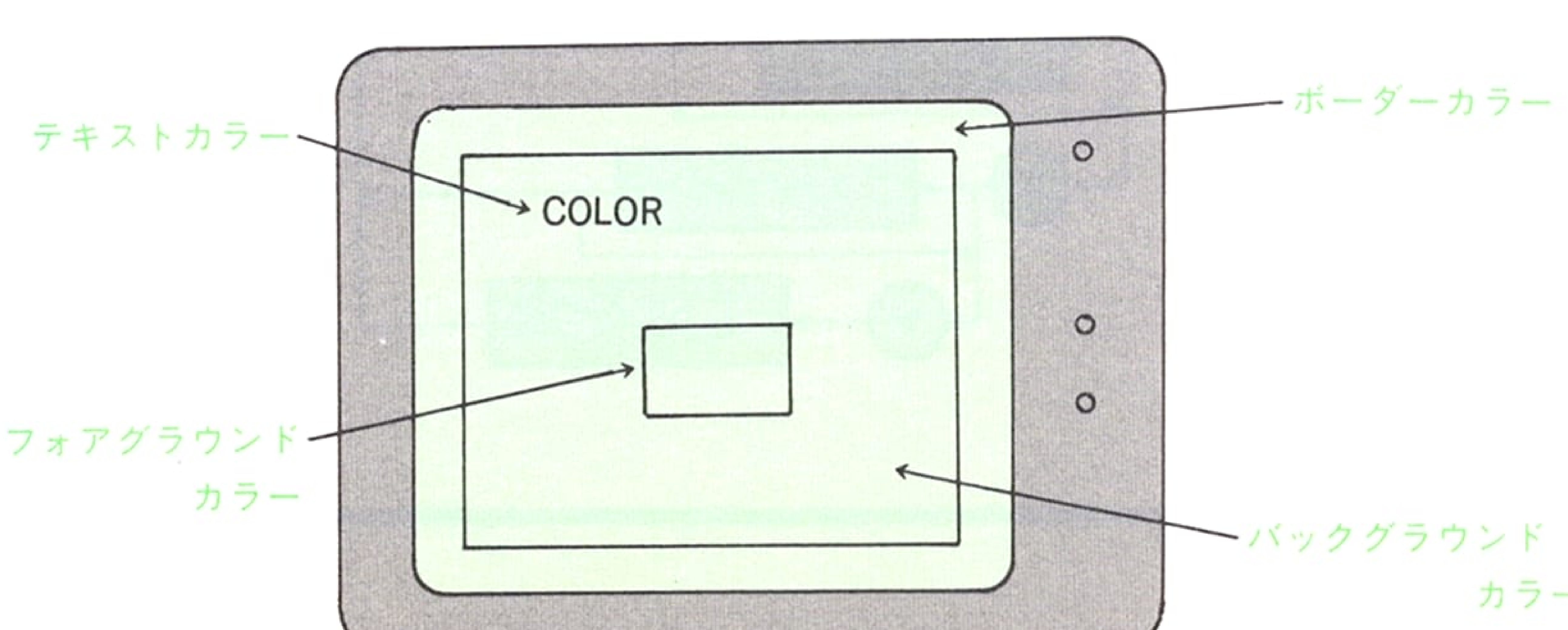
もう1つテキスト画面のカラー指定についてお話しておきましょう。テキスト画面に色をつけるのはあまり気がすすみませんが、機能的にはできることになっていますので、説明しておく必要があると思います。

テキスト画面に色をつけたい時にはCOLOR（カラー）文を使います。COLOR文には２種類ありますが、一方のCOLOR文を使うのです。COLOR文の構造図を見て下さい。テキスト画面の色づけに関係するのは、一番最初のファンクションコードだけです。このファンクションコードも場合わけがあります。つまりテキスト画面がカラーモードか白黒モードであるかによって場合がわかれるのです。

ファンクションコード	白黒モードの場合（CONSOLE,,,0）	カラーモードの場合（CONSOLE,,,1）
0	ノーマル——通常の表示	黒
1	シークレット——文字は表示されない	青
2	ブリンク——点滅	赤
3	シークレット——文字は表示されない	紫
4	リバース——反転する	緑
5	リバースシークレット——反転して文字は表示されない	水色
6	リバースブリンク——反転して点滅	黄
7	リバースシークレット——反転して文字は表示されない	白

文字が表示されないモードなどに入ってしまいますと大変困りますから、COLOR文の第１パラメータにはよほど気をつける必要があります。CONSOLE文の第４パラメータに何を指定したかなど、なかなか記憶できるものではないからです。逆にテキスト画面の文字がどうしても見えないという時には、次のように入力してやると復帰できる場合があることがわかります。

```
CONSOLE 0,25,0,0 ⏎
COLOR 0 ⏎
```



0·3　グラフィック画面の表示を決めること

グラフィック画面の指定にはSCREEN文を使います。SCREEN文の構造図はきわめて簡単ですが、この4つのパラメータの内容を完全に理解するには数10ページを要します。そこで煩わしいという方は640×200ドットのカラーCRTの場合には次のように指定するだけにして下さい。

```
SCREEN 0,0
```

これだけでも何とかなりますが、何事もきわめ尽くさないと心配だという方も多いでしょうから、2つのパラメータの説明をしておきましょう。第1番目のパラメータは画面のモードを指定するものです。

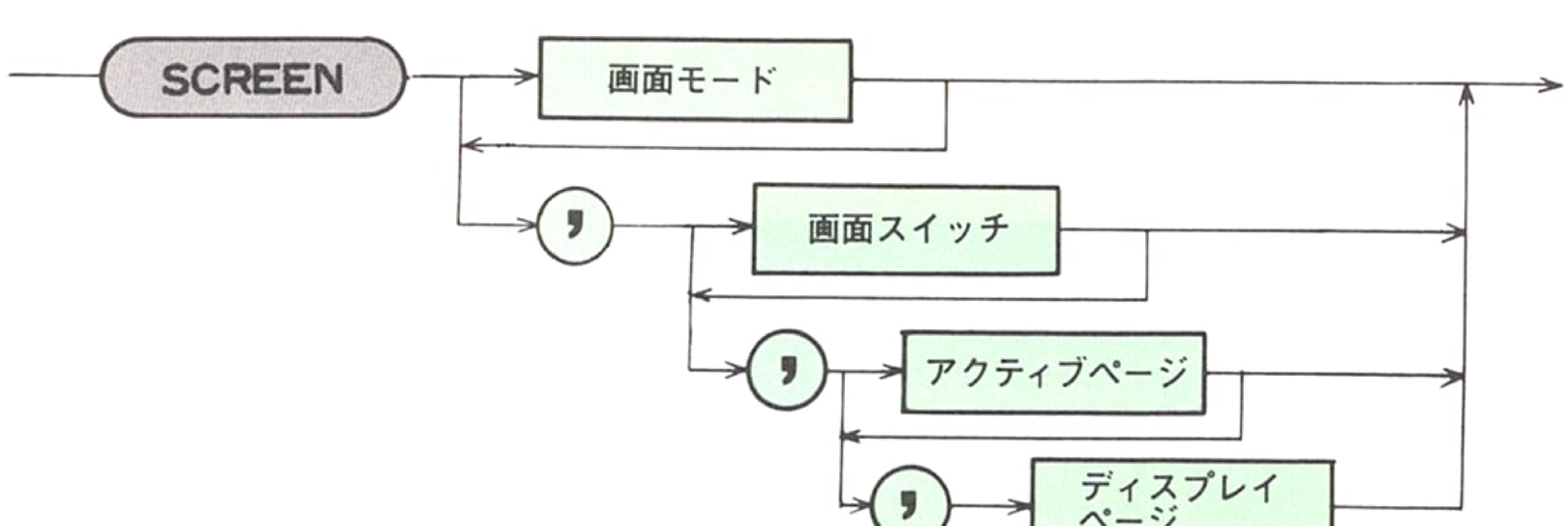
パラメータ	モード	ドット数・ページ数
0	カラーモード	640×200ドット、1ページ
1	白黒モード	640×200ドット、3ページ
2	高分解能白黒モード	640×400ドット、1ページ

SCREEN　●SCREEN／スクリーン

目的　グラフィック画面に対して種々のモードを設定する。

書式　SCREEN　〔〈画面モード〉〕〔,〈画面スイッチ〉〕〔,〈アクティブページ〉〕〔,〈ディスプレイページ〕〕

文例　SCREEN　1,0,0,7



第２番目のパラメータは画面スイッチをあらわします。画面スイッチには、グラフィックマスクスイッチと高速書き込みスイッチの２つがあります。

パラメータ	グラフィック画面マスクスイッチ	高速書き込みスイッチ	機　　能
0	OFF	OFF	グラフィック描画が普通の速度で行なえる
1	OFF	ON	グラフィック描画が不鮮明であるが高速にできる
2	ON	OFF	グラフィック描画が普通の速度でできる
3	ON	ON	グラフィック描画が高速にできる

画面の高速消去

画面を高速に消すテクニックとして640×200ドットの場合、

```
SCREEN 0,3: CLS 3: SCREEN 0,0 ↵
```

が使われることがあります。

第３、第４番目のパラメータについてはもう少し先へ行ってからお話することにしましょう。あまり完璧を期すと先へ進めません。

パソコン・グラフィックスにおいて特に力をおいて勉強すべきものとして，私は次の７つを考えています。

❶ SCREEN
❷ WINDOW
❸ VIEW
❹ COLOR
❺ STEP
❻ PAINT TILE$
❼ READ DATA

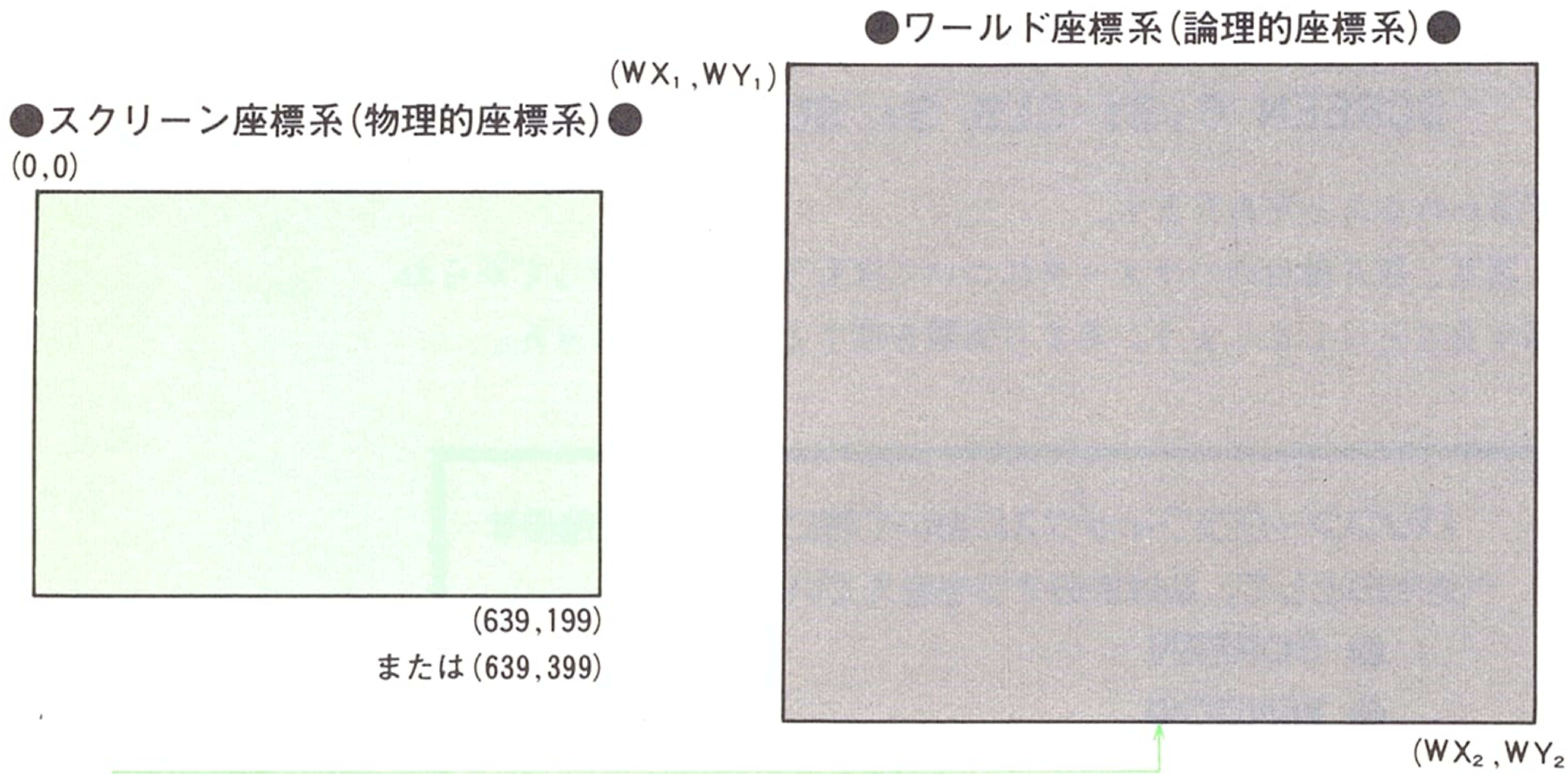
これにフロッピーディスクの使い方，ディジタイザーの使い方，プロッタの使い方をマスターすれば，かなりのことができるようになります。

0·4 座標系とは何だろう

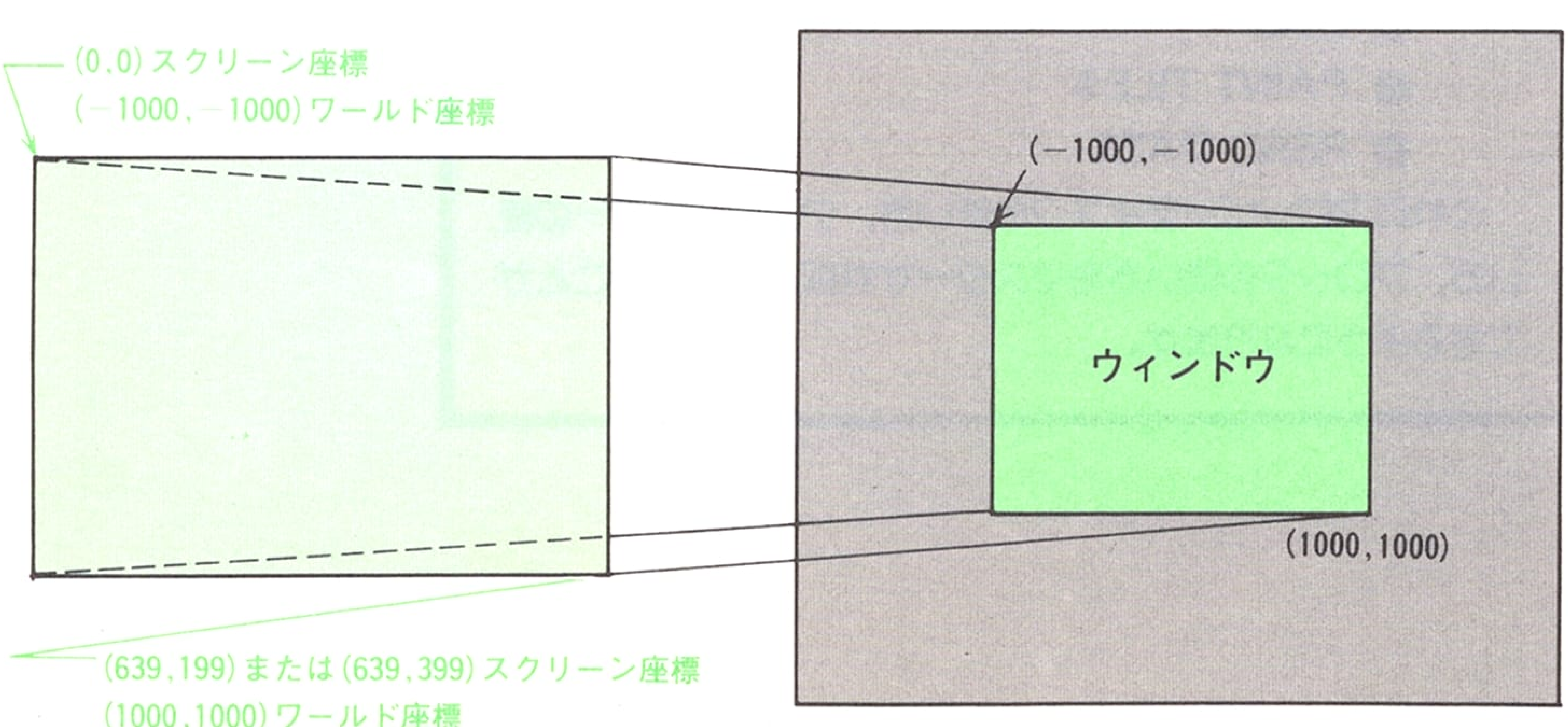
さて、これまでにテキスト画面とグラフィック画面表示を決定したのですが、最初からキチンと勉強しておいた方が良いと思うものが１つあります。それは座標系というものです。

スクリーン座標系
ワールド座標系

PC-８８０１のグラフィック画面ではスクリーン座標系とワールド座標系という２つの座標系が使えます。スクリーン座標系というのは画面上の座標系で実際の画面の１ドットずつとの１対１の対応を持っており、(0, 0) から (639,199) もしくは (0,0) から (639,399) という画面を構成します。

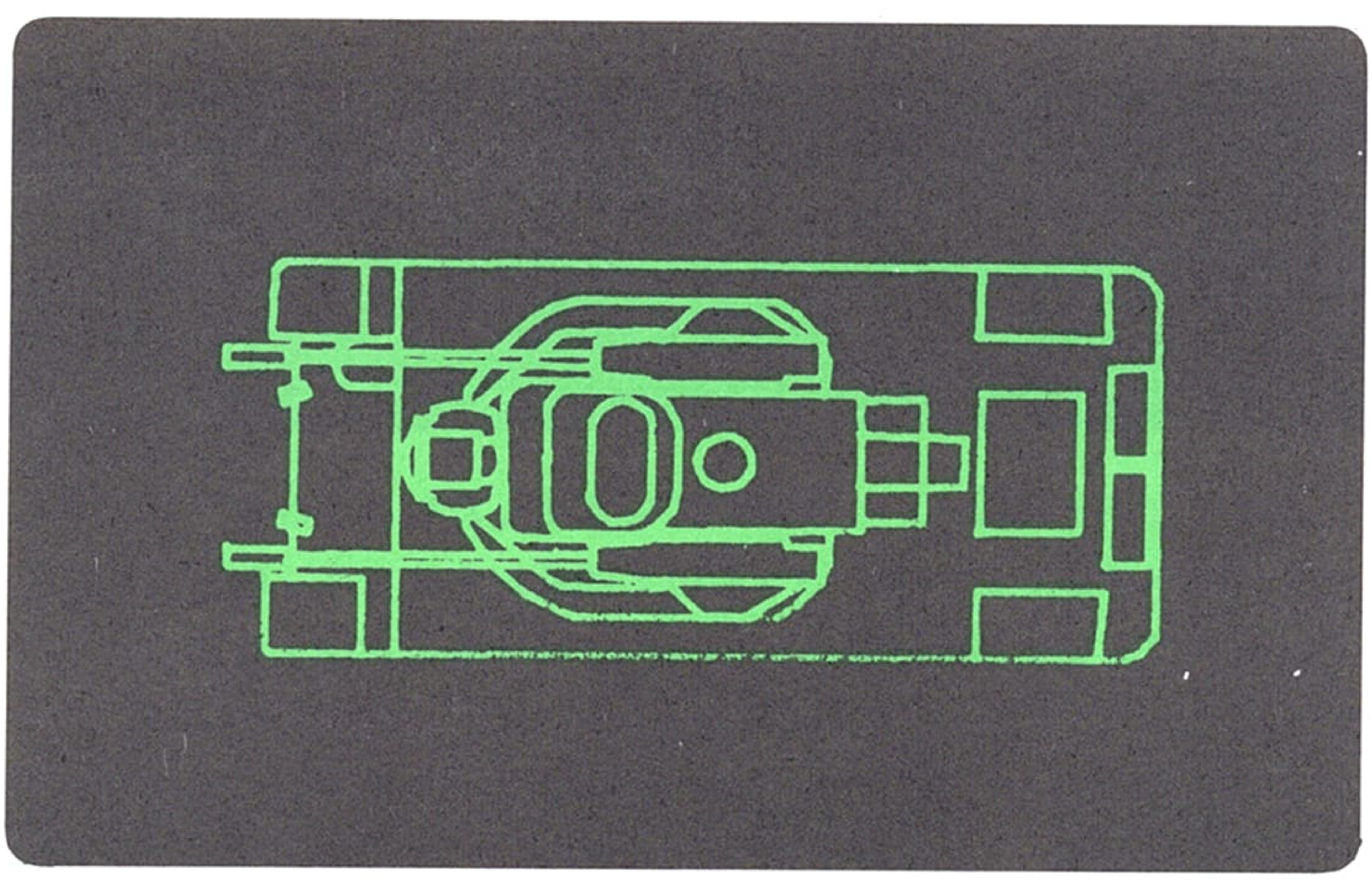


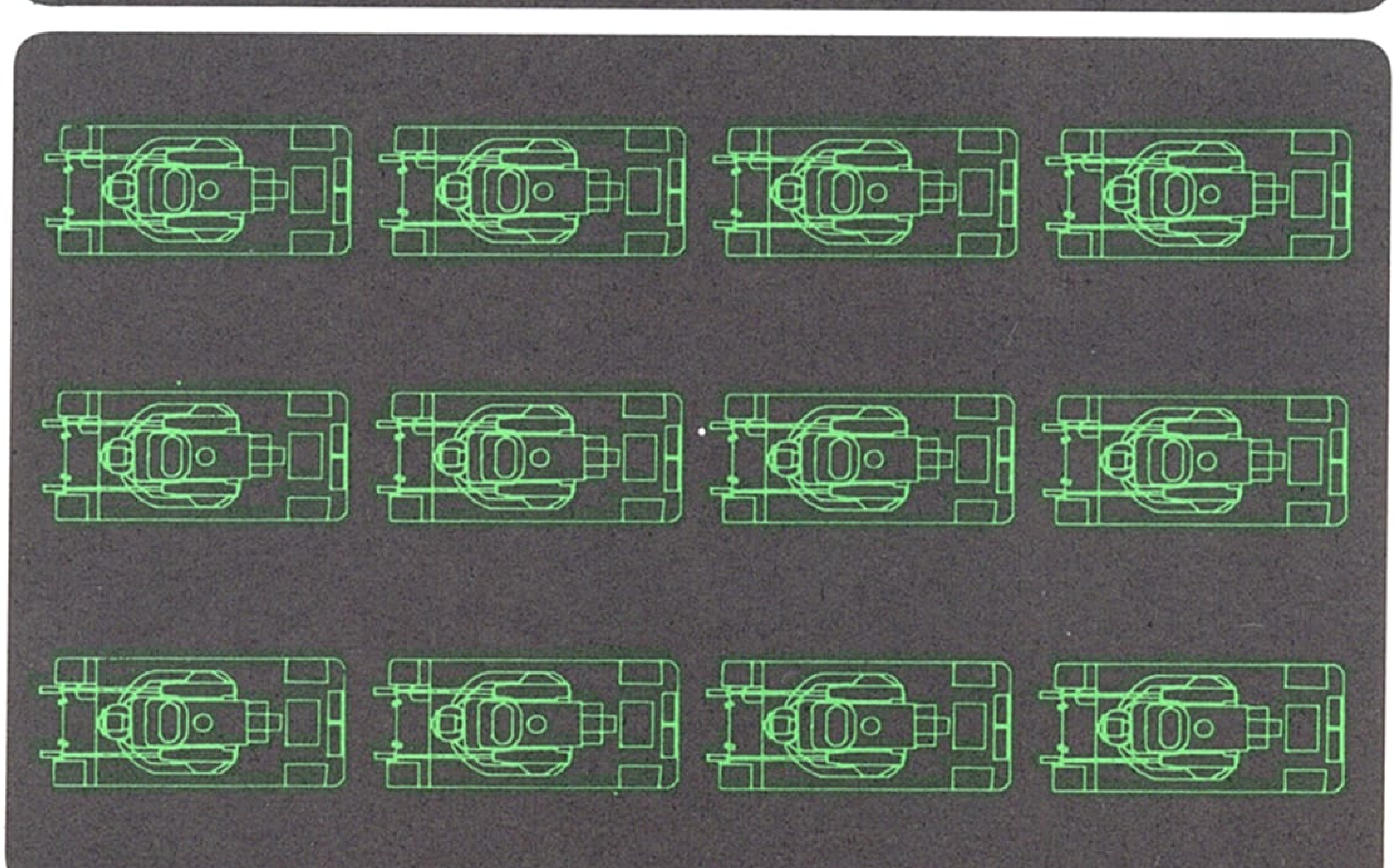
ただし、$WX_1 - WX_2 < 1.7041 \times 10^{38}$　$WY_1 - WY_2 < 1.7041 \times 10^{38}$
かつ、$WX_1 < 1.7041 \times 10^{38}$　$WX_2 < 1.7041 \times 10^{38}$



ワールド座標系はBASICが設定した仮想的な座標系で、きわめて広大な空間を持っています。ワールド座標系が使えることは大変便利なことなのです。たとえば、ディジタイザーという機械をPC-８８０１につなげることを考えてみますと、この場合ディジタイザーは数千ドット×数千ドットの空間を持っており、これと画面の640×200ないし640×400ドットの空間は大きさが違いますから、画面上にディジタイザーからの入力図形を表示しようとすると、縮小したり拡大したり、切り出したりしなければなりません。するとどうしても座標の変換が必要になります。見たい図形を画面にキチンと出すことはなかなか厄介ですが、ワールド座標系が使えればWINDOW文によって座標変換がきわめて楽にできることになるのです。

たとえば、座標系の変換を上手に使いますと、もともとは１つの図形しか入力していない場合でも、多数の図形を表示することができるようになるのです。論より証拠といいますから次の図を見て下さい。





0・5　グラフィック画面のしくみ

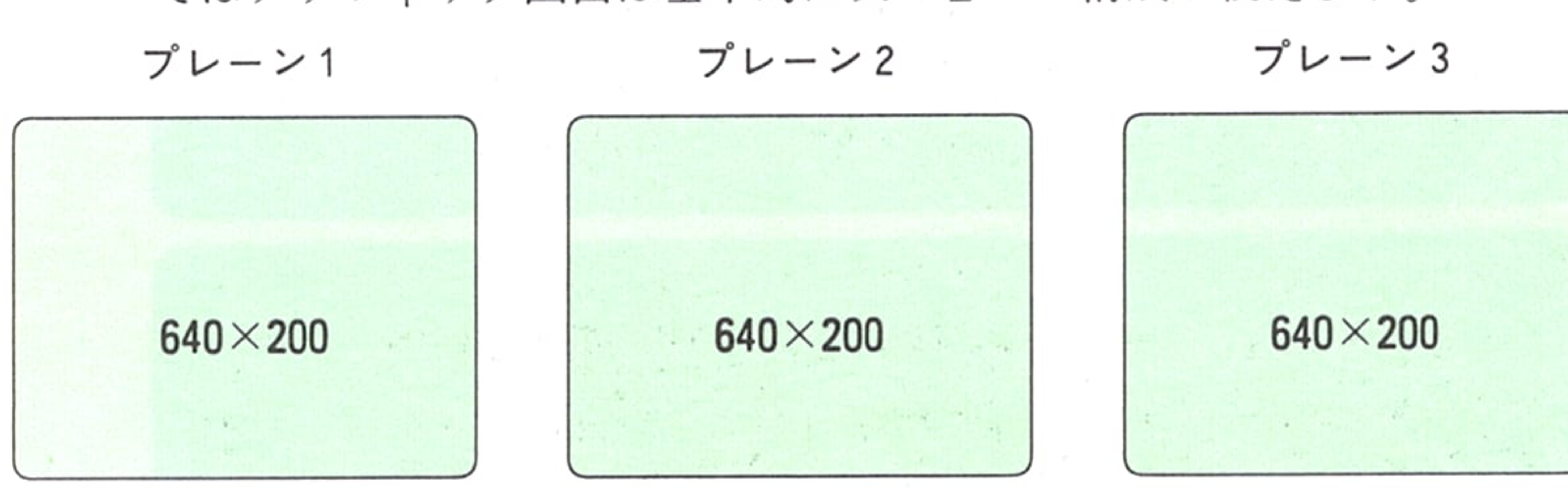
PC－８８０１で自由自在にグラフィックスをやるためには、グラフィック画面のしくみに良く通じていなければなりません。画家がキャンバスのことを知らずに絵を描けないのと同じです。

グラフィック画面	
分解能（横×縦）	備　　考
640×200	カラー１ページまたは白黒３ページ
640×400	白黒１ページ

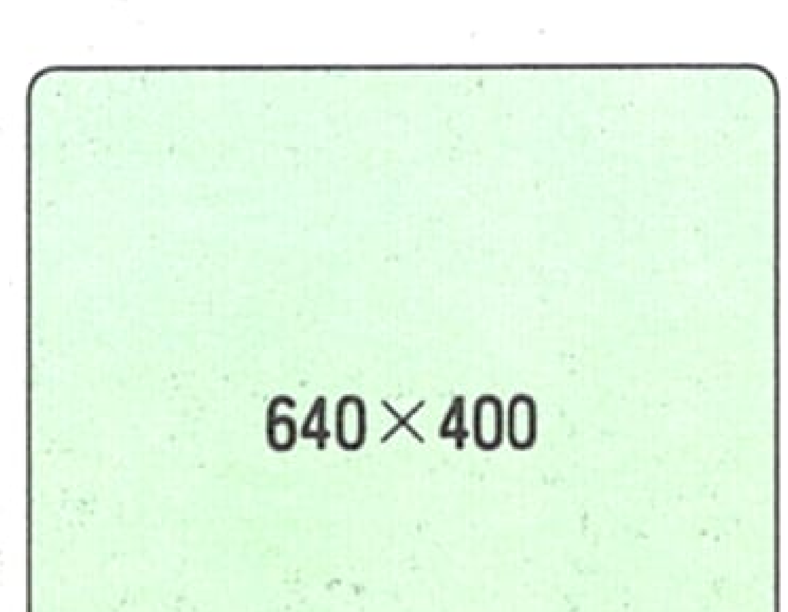
グラフィック画面についてはPC－８８０１のマニュアルに上のような説明があります。これでも良いのですが、次のように分類すると、もっとわかりやすくなります。

分解能（横×縦）	カラー/白黒	使えるページ数	BASICでの指定法
640×200	カラー	1	SCREEN 0
640×200	白黒	3	SCREEN 1
640×400	白黒	1	SCREEN 2

ここで、ページ数というものがでてきましたね。これは一体何でしょう。ページはプレーンというものと関係があります。PC－８８０１ではグラフィック画面は基本的に次の２つの構成が使えます。



640×200のプレーン　●カラー１ページまたは白黒３ページの場合



640×400のプレーン　●白黒１ページの場合

良く見ているとすぐわかるように640×200の場合３枚のプレーンが、640×400の場合１枚のプレーンが使えます。

これを利用するとグラフィックを有効にすることができます。

白黒モードの場合
(640×200ドット)

たとえば白黒の640×200モードの場合を見てみましょう。この場合３枚のプレーンが使え、３枚のプレーンと３ページは同一に考えて良いのです。つまり３枚の画面を使えますので、３枚の画面に別々の絵を描いて、それを交互に切りかえて表示することもできるのです。例題を１つ作ってみました。

●プログラム０－１●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,0
120 '
130 PI=3.1416
140 FOR I=0 TO 2
150 TH=PI/9*I
160 SCREEN 1,0,I,2^I
170 FOR J=1 TO 6
180 GOSUB *MCIRCLE
190 NEXT J
200 NEXT I
210 '
220 FOR I=0 TO 1000
230 FOR J=1 TO 50:NEXT J
240 K=I MOD 3
250 SCREEN 1,,,2^K
260 NEXT I
270 '
280 END
290 '
300 *MCIRCLE
310 WINDOW (0,0)-(639,400)
320 PX=80*J:PY=100
330 CIRCLE(PX,PY),36
340 CIRCLE(PX,PY),40
350 FOR N=0 TO 5
360 X=25:Y=0
370 W=X*COS(N*PI/3+TH)-Y*SIN(N*PI/3+TH)
380 Y=X*SIN(N*PI/3+TH)+Y*COS(N*PI/3+TH)
390 X=W
400 CX=PX+X:CY=PY+Y
```

```
410 CIRCLE(CX,CY),10
420 NEXT N
430 PAINT(PX,PY)
440 RETURN
```

●プログラム0—2●

```
100 '***** PROGRAM 0-2 *****
110 '
120 SCREEN 0,3:CLS 3:SCREEN 0,0
```

●画面を消すプログラム
これを使わないと0−1のプログラムの実行後画面をきれいにできない。

アクティブページ

SCREEN文の**第3パラメータ**はアクティブのページと言ってグラフィックを書き込むページをあらわします。白黒の640×200の場合、0〜2までの3ページがこれに該当します。

ディスプレイページ

SCREEN文の**第4パラメータ**はディスプレイページといって、CRT画面上に表示するページをあらわします。このプログラムでは1,2,4だけを選んでいます。これらの値を選ぶ理由は下の表をみていただけばわかるでしょう。

ディスプレイページの値	意味	画面モードでの使用可否		
		0	1	2
0	全ページ表示しない	○	○	○
1	ページ1のみ表示	○	○	○
2	ページ2のみ表示	○	○	○
3	ページ1とページ2を合成して表示	×	○	×
4	ページ3のみ表示	×	○	×
5	ページ1とページ3を合成して表示	×	○	×
6	ページ2とページ3を合成して表示	×	○	×
7	ページ1とページ2とページ3を合成して表示	×	○	×

●SCREEN文の第4パラメータ、ディスプレイページの値

カラーモードの場合
(640×200ドット)

640×200の場合、カラーにすると事情が少し異なってきます。カラー化する方法としてR(赤)、G(緑)、B(青)の3枚のプレーンを使って光の3原色によっています。だから、3枚のプレーンによって1つの画面すなわち1ページができます。640×200の場合、3枚のプレーンがあるわけですから、カラーにすると、ちょうど1ページが使えます。

SCREEN文の第3パラメータは0が使えます。第4パラメータは0,1,2しか使えません。第3パラメータと第4パラメータの使い方は、かなり覚えるのが厄介です。失敗をくり返して身につけていく他はありません。これも1つ例題を作ってみましょう。



ページ1

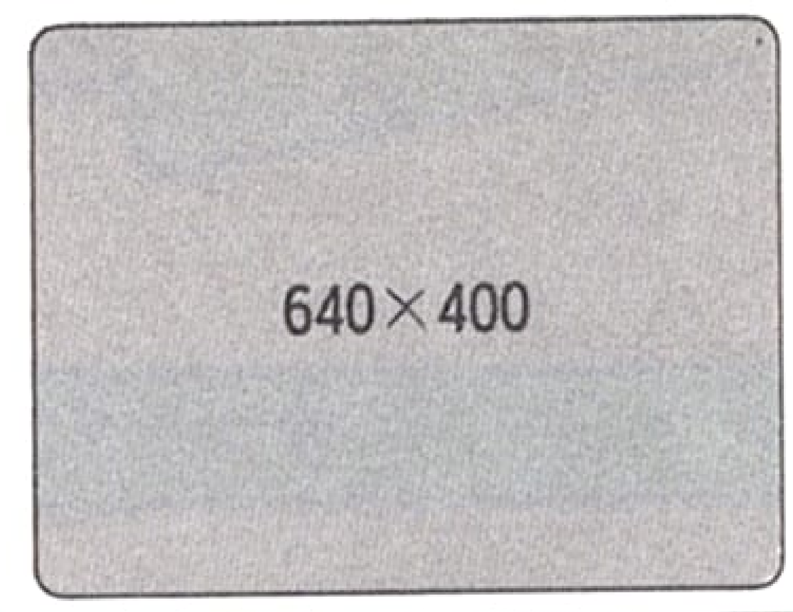
●プログラム0―3●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 '
130 PI=3.14159
140 FOR I=0 TO 2
150 TH=PI/9*I:SCREEN 1,0,I,2^I
160 FOR J=1 TO 7:GOSUB *MCIRCLE:NEXT J
170 NEXT I
180 '
190 SCREEN 0,0
200 LINE(25,135)-(70,120),7
210 LINE-(580,120),7:LINE-(610,135),7
220 LINE(100,120)-(200,80),7:LINE-(300,80),7
230 LINE-(300,120),7
240 LINE(300,105)-(500,105),7:LINE-(540,120),7
250 LINE(220,80)-(220,75),7
260 LINE-(275,75),7:LINE-(275,80),7
270 LINE(100,120)-(120,90),7
280 LINE-(150,85),7:LINE-(190,85),7
290 '
300 FOR J=1 TO 100
310 FOR I=0 TO 2:SCREEN 1,0,I,2^I:NEXT I
320 NEXT J:END
330 '
340 *MCIRCLE
350 WINDOW(0,0)-(639,399)
360 PX=80*J:PY=300
370 CIRCLE(PX,PY),36:CIRCLE(PX,PY),40
380 FOR N=0 TO 5
390 X=25:Y=0
400 W=X*COS(N*PI/3+TH)-Y*SIN(N*PI/3+TH)
410 Y=X*SIN(N*PI/3+TH)+Y*COS(N*PI/3+TH)
420 X=W:CX=PX+X:CY=PY+Y
430 CIRCLE(CX,CY),9
440 NEXT N
450 PAINT(PX,PY)
460 RETURN
```

高分解能白黒モードの場合（640×400ドット）

640×400の白黒モードの場合、これはもう原理がおわかりでしょうが、1枚のプレーンが使えます。640×200が2枚あると640×400が1枚できます。

この場合、気をつけることは第3パラメータが0，1，以外の値をとれないこと、第4パラメータが0から3までしか使えないことだけです。



高分解能白黒モード(1ページ)

色	緑のプレーン（ビット2）	赤のプレーン（ビット1）	青のプレーン（ビット0）	カラーコード
黒	0	0	0	→ 0
青	0	0	1	→ 1
赤	0	1	0	→ 2
紫	0	1	1	→ 3
緑	1	0	0	→ 4
水色	1	0	1	→ 5
黄色	1	1	0	→ 6
白	1	1	1	→ 7

3ビットの2進数とみると右のカラーコードの数値になる。

あまり難しい話はさけようと思いますが、RGB(**R**ed, **G**reen, **B**lue)の3つのプレーンをどう使っているかを示しているのが右の図です。

カラーコードは覚えにくいものですが、私は勝手に次のような暗記コードを作って覚えています。

黒　(0)――黒いレイ服
青　(1)――青春はイチ度
赤　(2)――赤いニンジン
紫　(3)――紫はミダ三尊
緑　(4)――緑ショウ
水色(5)――水色のコイ
黄　(6)――六本ギ
白　(7)――ナナ色混ぜると白

抵抗のカラーコードと混じる部分もありますが、役には立つと思います。PC-８８０１でカラー指定をする上においてはWINDOW 文やVIEW文、さらにPALETTE文という強力な命令群を良く理解しておく必要がありますので、例題のプログラムをここでとり上げることはせず、先へ進むことにします。

ここで、サンプルプログラムでたびたび出てきたCLSについてふれておきます。

これは画面をクリアする命令で、〈機能〉は1,2,3の値をとります。省略された場合は1が選択されます。

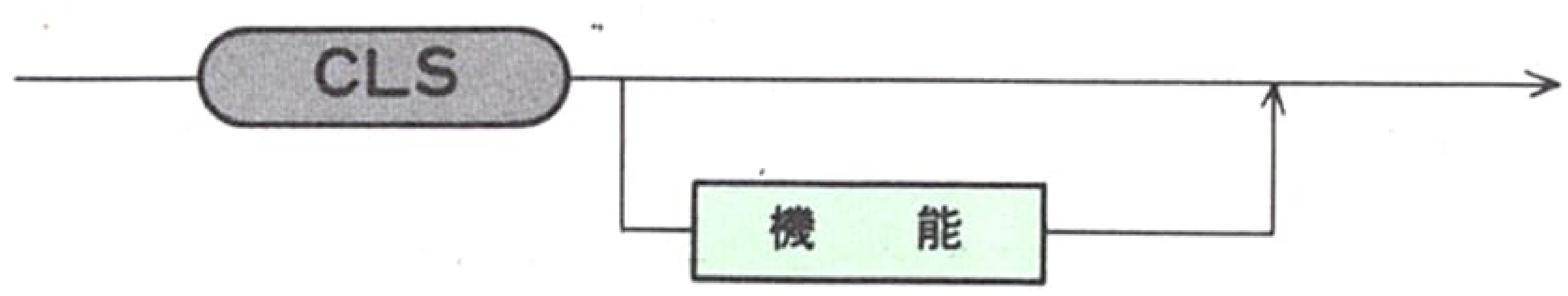
パラメータが1の時——テキスト画面のみクリアします。テキスト表示モードがCONSOLE文、COLOR文により、白黒リバースモードになっている場合には画面は白くなります。

CLS ●clear screen／クリア・スクリーン

目的 現在アクティブな画面をクリアする

書式 CLS 〔〈機能〉〕

文例 CLS 2

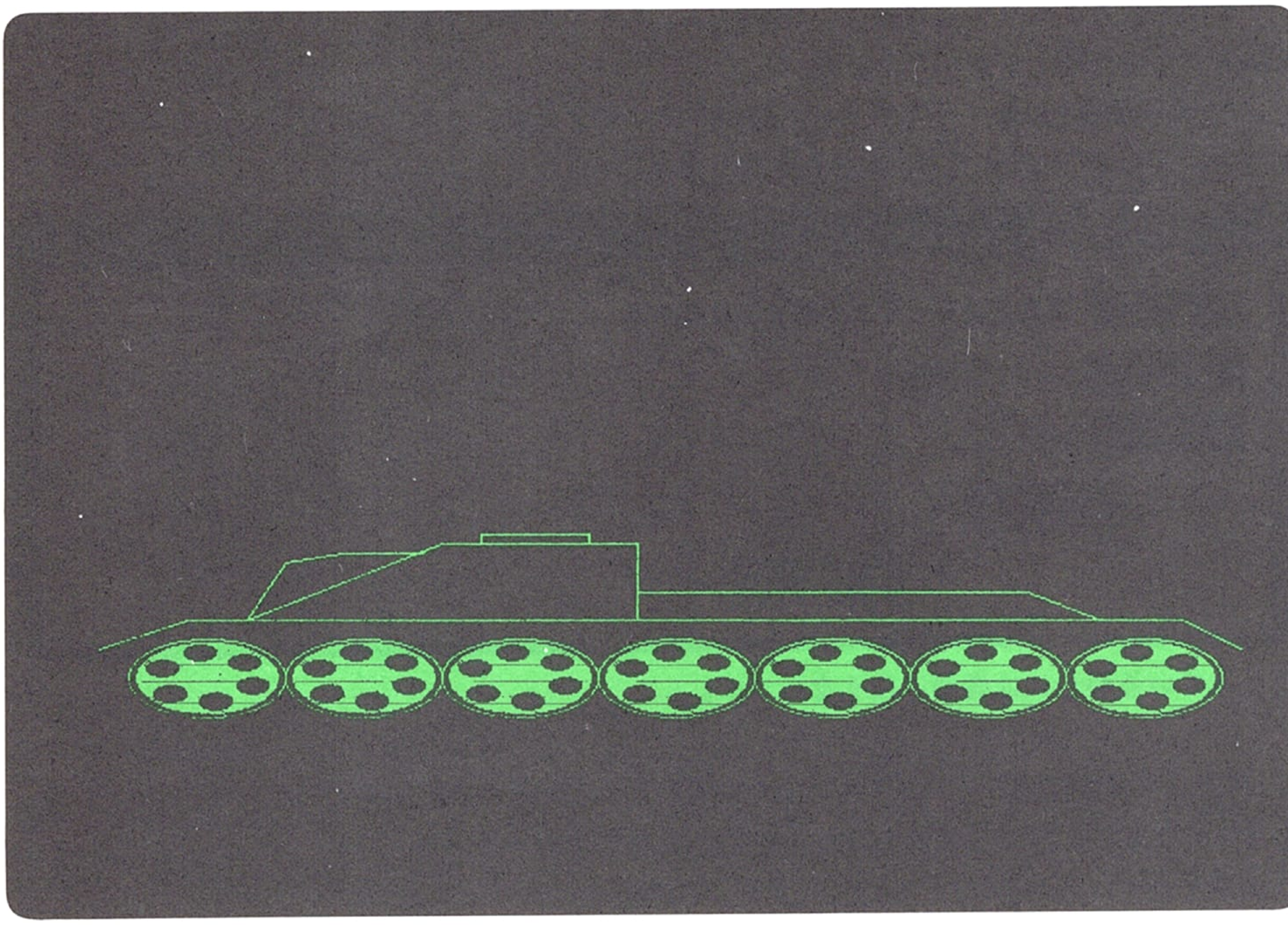


パラメータが2の時——グラフィック画面のビューポート内のみをクリアします。グラフィック表示がSCREEN文によりカラーモードに設定されている場合には、COLOR文により指定されたバックグラウンドカラーで、ビューポート内をクリアします。

パラメータが3の時——テキスト画面、グラフィック画面の両方をクリアします。

テキスト画面がクリアされる時は、CONSOLE文によって指定されたスクロールウィンドウ内のみをクリアします。またグラフィック画面をクリアした場合には、LP(Last referenced Point)はビューポートの左上の頂点に移動します。

次に示すのは、この節であつかったプログラムを実行したときの一画面です。



このグラフィックは蒸気機関車の動輪や装軌車両の可動部分をモデルに考えて作っています。本来は7個の動輪か、不均等な位置に配置された動輪が望ましいのですが、上部構造物を作らなかったのでこのままにしてあります。SLファンや模型のファンの方はアイデアを生かして活用されるとよいでしょう。本当に動きだすSLを画面に作りだすと面白いですよ。

0・6　カラー・パレットとは何だろう

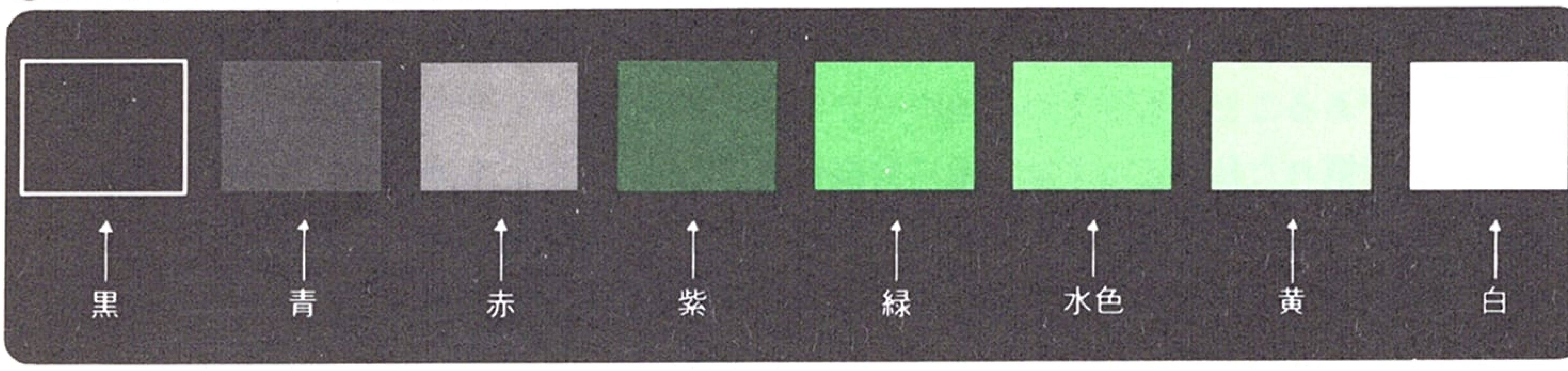
PC-８８０１とPC-９８０１において特に悩まされるのがカラーパレットです。悩まされるというのは２つの意味があって、１つは理解しにくい所があること。もう１つは他のパソコンではやりにくいことで、FM-８で慣れた私にとっては二重に負担を感じていました。しかし今やPC-８８０１を自在に使えるのですから、パレット文と同じ動作をさせるプログラムをどう作るべきかなどと悩む必要はないわけです。パレットとは絵を描く時のパレットと同じと説明されています。しかし少し感じが違います。むしろ一度塗ってしまった色を一瞬にしてかえることのできる命令であると理解した方がよいでしょう。実際にプログラムを作って理解しましょう。

●プログラム０－４●

```
100 '*****  PROGRAM 0-4  *****
110 '
120 WIDTH 80,20:CONSOLE 0,20,0,0            ┐
130 SCREEN 0,3:CLS 3:SCREEN 0,0,0,1         ┘― 画面の初期設定
140 '
150 X=50:Y=25:GOSUB *CPAINT  ←―― 7色の箱を作る
160 X=50:Y=25:GOSUB *CBOX  ←―― 箱の枠を作る
170 '
180 FOR I=1 TO 7:COLOR=(I,1):NEXT I ←― パレットの色を青にする
190 FOR J=1 TO 1000:NEXT J ←―― タイミング
200 FOR I=0 TO 7:COLOR=(I,I):NEXT I ←パレットの色を元にもどす
210 '
220 END
230 '
240 *CBOX                                   ┐
250 FOR I=0 TO 7                            │
260 POINT(X,Y)                              │
270 LINE STEP(0,0)-STEP(40,25),7,B          ├ 箱の枠を作る
280 X=X+50                                  │     サブルーチン
290 NEXT I                                  │
300 RETURN                                  ┘
310 '
320 *CPAINT                                 ┐
330 FOR I=0 TO 7                            │
340 POINT(X,Y)                              │
350 LINE STEP(0,0)-STEP(40,25),I,BF         ├ 7色の箱を作る
360 X=X+50                                  │     サブルーチン
```

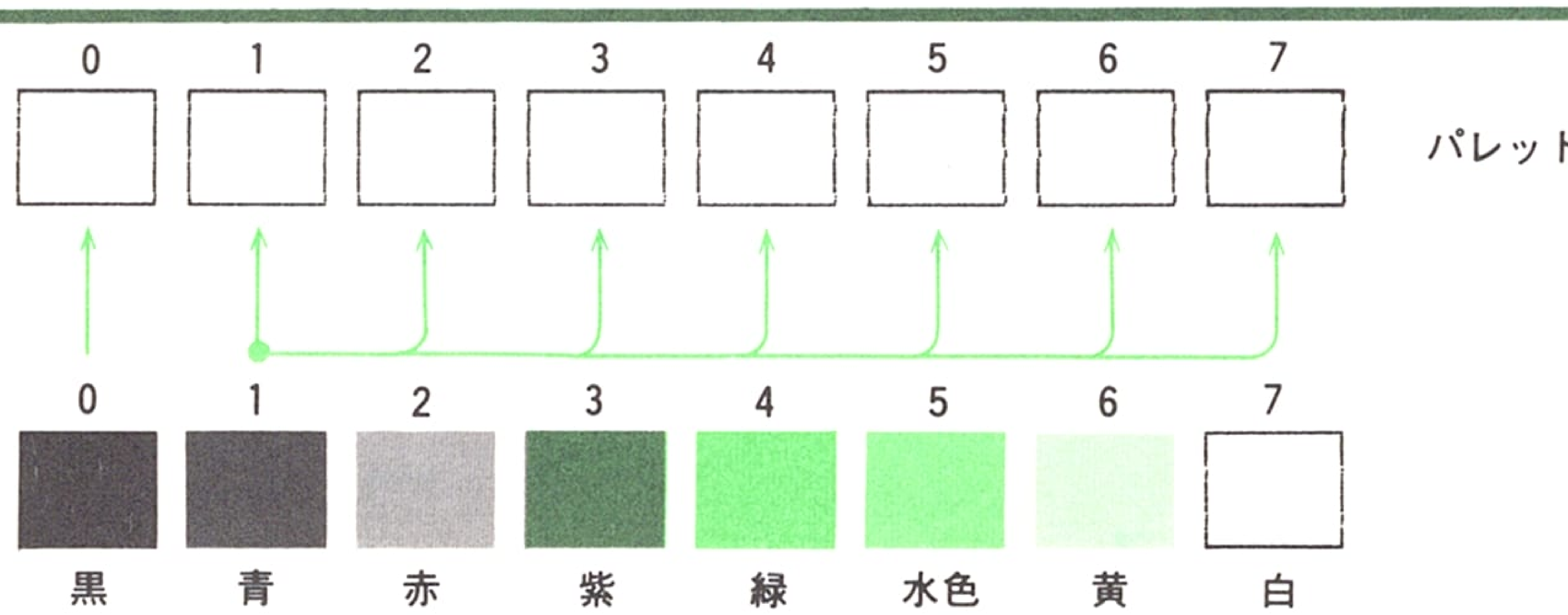
```
370 NEXT I
380 RETURN
```

●プログラム１－４の実行結果



プログラム０－４では、７つの箱を作って色を塗っています。電源を入れてから特に何もしない限りは、パレットにはカラーコード通りの色が入っています。つまり０番のパレットには０番の絵具である黒が、１番のパレットには１番の絵具である青が、２番のパレットには２番の絵具である赤がというように、しごくふつうのやり方で絵具がしぼり出されています。ですから箱を塗るサブルーチンではカラーコード通りの色が塗られています。

行番号180の COLOR 文でパレットの色の入れかえが行なわれています。COLOR＝(I,1)はＩ番目のパレットに１番目の絵具である青をしぼりだしているのです。つまり絵具のパレットを洗って、絵具のチ



●パレットの０～７の任意の場所に絵具をしぼり出せる。
●プログラム０－４の180行では１～７番のパレットに青をしぼり出している。

ューブから新しいやり方でパレットにしぼり出しているのです。行番号180の FOR～NEXT ループでＩを１から７まで回し、０を通さないのは、画面のバックグランドカラーまでが変化してしまい、画面が真青になって面白味を欠いてしまうからです。

行番号 190 はタイミングをとるものです。これを入れておかないと色が変化するのが眼に止まりません。

行番号 200 はもう一度カラーパレットを初期状態にもどしておくも

のです。こうしておかないと、プログラムを新しくした場合、いつでも青で表示されてしまい原因がわからなくなってしまいます。

プログラムの中にはまだ全然説明していない技巧が使われています。たとえばPOINT文やLINE STEP文などは説明していません。今はそういうものだと思って下さい。後でだんだん説明を加えて行きます。

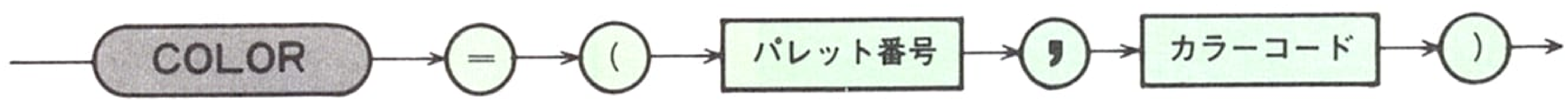
カラーパレットを使ったアニメーションのテクニックもあります。

COLOR ●color／カラー

目的 カラーパレットを変更する

書式 COLOR＝(〈パレット番号〉,〈カラーコード〉)

文例 COLOR＝(2, 4)



〈パレット番号〉	〈カラーコード〉
0	0 (黒)
1	1 (青)
2	2 (赤)
3	3 (紫)
4	4 (緑)
5	5 (水色)
6	6 (黄色)
7	7 (白)

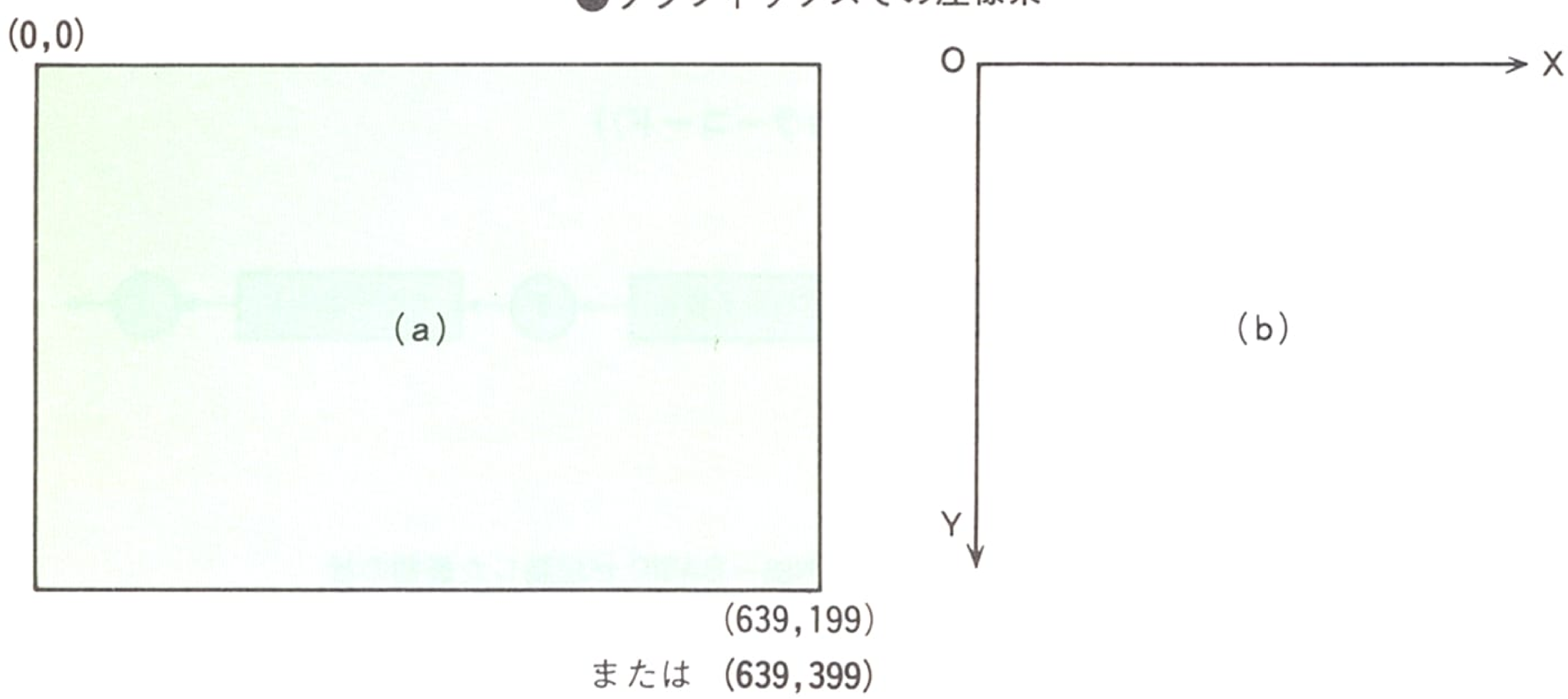
N88－BASICが起動した最初の状態は、このように〈パレット番号〉＝〈カラーコード〉の対応となっているが、COLOR文によって、どのパレットにどのカラーコードを対応させるか任意にきめることができる。

0・7　座標変換とグラフィックス

座標変換

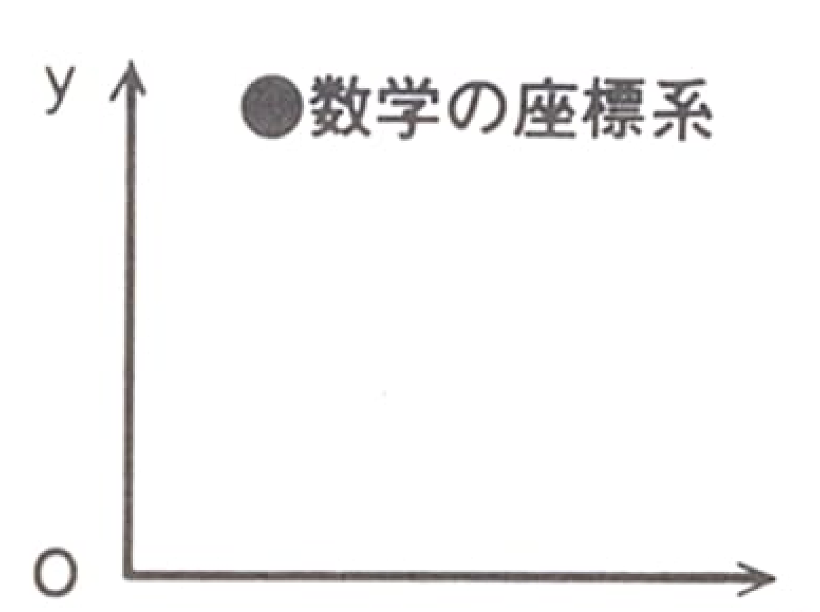
座標系というやつは全く厄介です。しかし、グラフィックスを上手にやろうとすれば、座標系は絶対に必要です。そして**座標変換**という操作が必要になってくるのです。数学で座標変換を習った人は、ああ、あの面倒なやつかと思われるかも知れませんが、これまでにも皆さんに座標変換を使っています。たとえばスクリーンにおける座標系は次のように決められていました。

●グラフィックスでの座標系



これは数学で習った座標軸のとり方と少し違います。わかりやすいように座標軸だけ抜き出してみますと図(b)のようになります。Y軸のとり方が変でしょう。数学で使っていた座標系は、Y軸の正の部分が上にきていたのです。少しでもグラフィックを使っていた人は、知らず識らずのうちに、

●数学の座標系



$$\begin{cases} X = x \\ Y = -y \end{cases}$$

という座標変換をやっていたのです。

何か絵を画面に描こうとする時、原点Oがスクリーンの左はじにあるのはいやだと思いませんか。たとえば次の式で円を描こうと考えたとします。

$$x = 100 \times \sin(2\pi i/200)$$
$$y = 100 \times \cos(2\pi i/200) \qquad (i = 0 \text{ から } 200)$$

良く間違えるのは次のようなプログラムです。このプログラムの欠点はどこにあるでしょうか。少し考えてみて下さい。

●プログラム0－5●

```
100 CONSOLE 0,25,0,0
110 SCREEN 0,3:CLS 3:SCREEN 0,0
120 '
130 PI=3.14159
140 FOR I=0 TO 200
150 X=100*SIN(2*PI*I/200)
160 Y=100*COS(2*PI*I/200)
170 PSET(X,Y),4
180 NEXT I
190 '
200 END
```

← 0°～90°（0～π/4ラジアン）の間はプロットされるが、それを越えると画面からはみ出してしまう。

多分おわかりになったと思いますが、Y軸のとり方が良くないのです。円は上下左右対称ですから出来上る図形は同じですが、書き始めの位置が上下ひっくり返っています。Y軸のとり方だけについて言えば、次のようなプログラムの方が良いのです。

●プログラム0－6●

```
100 CONSOLE 0,25,0,0
110 SCREEN 0,3:CLS 3:SCREEN 0,0
120 '
130 PI=3.14159
140 FOR I=0 TO 200
150 X=100*SIN(2*PI*I/200)
160 Y=-100*COS(2*PI*I/200)
170 PSET(X,Y),4
180 NEXT I
190 '
200 END
```

← Y軸方向の値にマイナスをつける。しかし、この場合にも3π/4～π ラジアンの間だけがプロットされ、結果は前のプログラムと変わらない。

しかしこのプログラムにも欠点があります。Y軸方向が2倍で、画面の左上すみに4分の1の円しか出ないことです。ふつうは画面の中央に円を出したいですね。

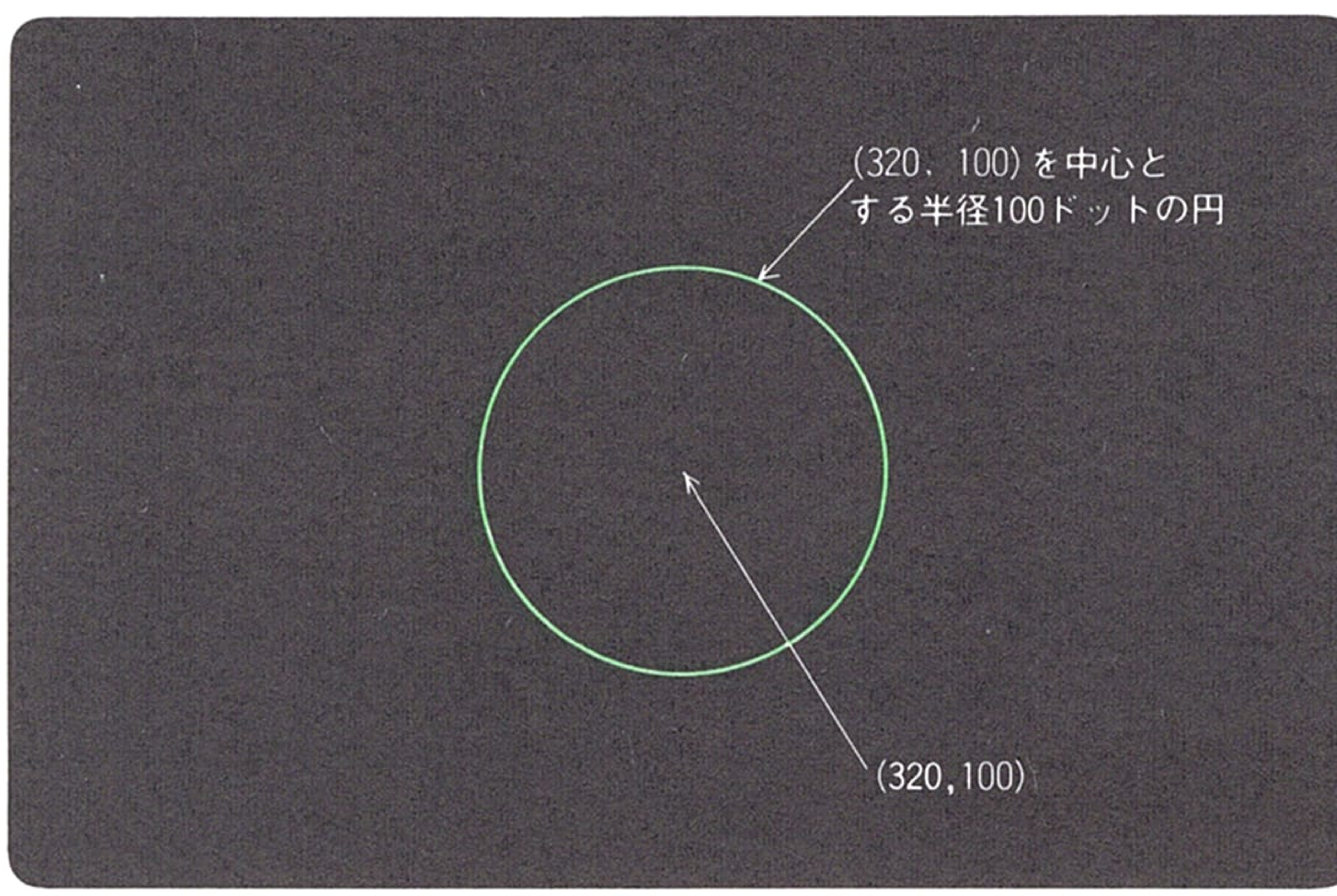
そこで原点を（320,100）にうつしてやりたくなります。そのためのプログラムが次のプログラムです。これできちんと円が画面中央に表示されることになります。



●プログラム０－７●

```
100 CONSOLE 0,25,0,0
110 SCREEN 0,3:CLS 3:SCREEN 0,0
120 '
130 PI=3.14159
140 FOR I=0 TO 200
150 X=100*SIN(2*PI*I/200)+320
160 Y=-50*COS(2*PI*I/200)+100
170 PSET(X,Y),4
180 NEXT I
190 '
200 END
```

（150・160行）原点を(320,200)に移動する



つまりもとのxy座標系とスクリーンの座標系を結ぶには、

X＝x＋320
Y＝－y＋100

としなければならないのです。これが混乱のもとになりがちです。グラフィックが表示されなかったり、エラーがでるのは殆んどすべて、この座標変換の失敗によります。

0・8 WINDOWとVIEWのやさしい使い方

さて我々はせっかく強力なWINDOW文を持っているのですからこれを利用しない手はありません。WINDOWは広大なワールド座標にかかれたグラフィックに窓かけをしてスクリーン座標系に表示するものです。我々の例では半径100の円を描いていますので、ちょっと考えると（－200，－200）と（200，200）を対角線の端点とすれば画面上に円がかけるように思います。ところがこれは失敗します。円が円にならないのです。プログラムをご覧下さい。

現在画面は640×200のモードにSCREEN文によって設定されています。だから（－200，200）と（200，200）を端点とする窓で良いように思えるのですが、画面に出てくるのは横長の楕円です。

この失敗の原因を反省してみる必要があります。失敗の原因は正方形の窓を640×200の長方形の画面に対応させてしまったことにありま

●プログラム0－8●

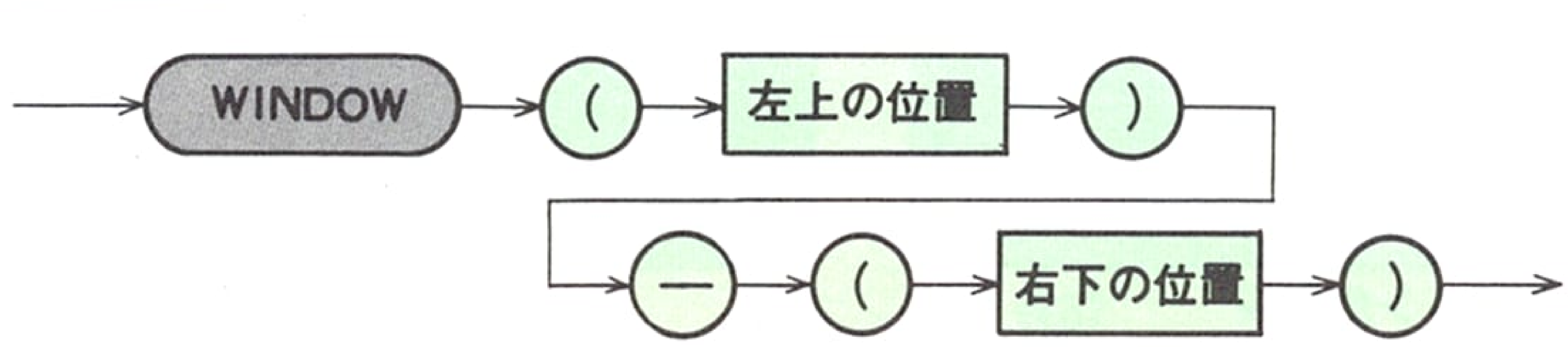
```
100 CONSOLE 0,25,0,0
110 SCREEN 0,3:CLS 3:SCREEN 0,0
120 WINDOW (-200,-200)-(200,200)
130 '
140 PI=3.14159
150 FOR I=0 TO 200
160 X=100*SIN(2*PI*I/200)
170 Y=-100*COS(2*PI*I/200)
180 PSET(X,Y),4
190 NEXT I
200 '
210 END
```

WINDOW ●window／ウィンドウ

目的 ビューポートに表示するワールド座標系内の領域を指定する

書式 WINDOW $(WX_1, WY_1)-(WX_2, WY_2)$

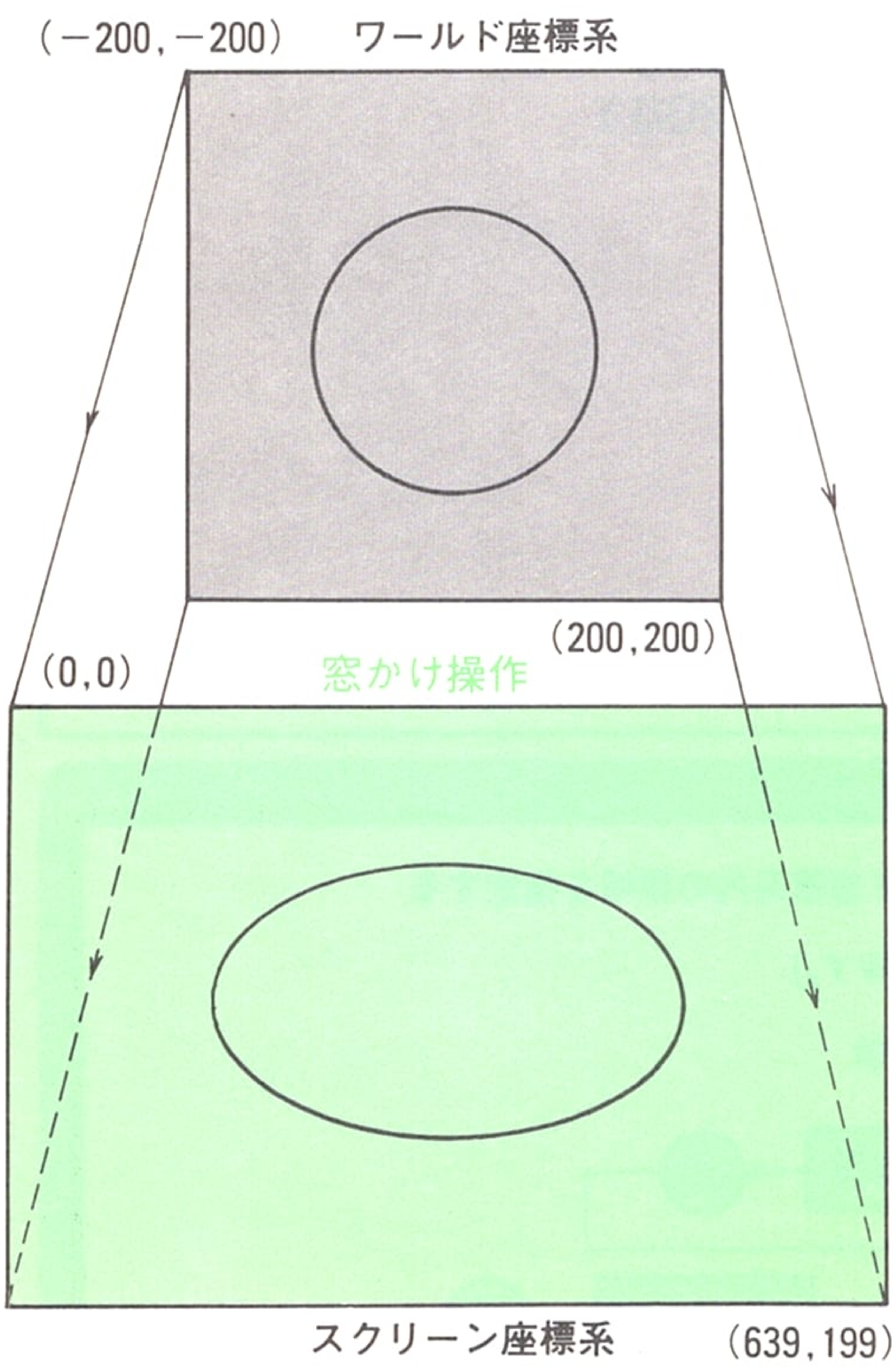
文例 WINDOW (－300, －50)－(100, 70)



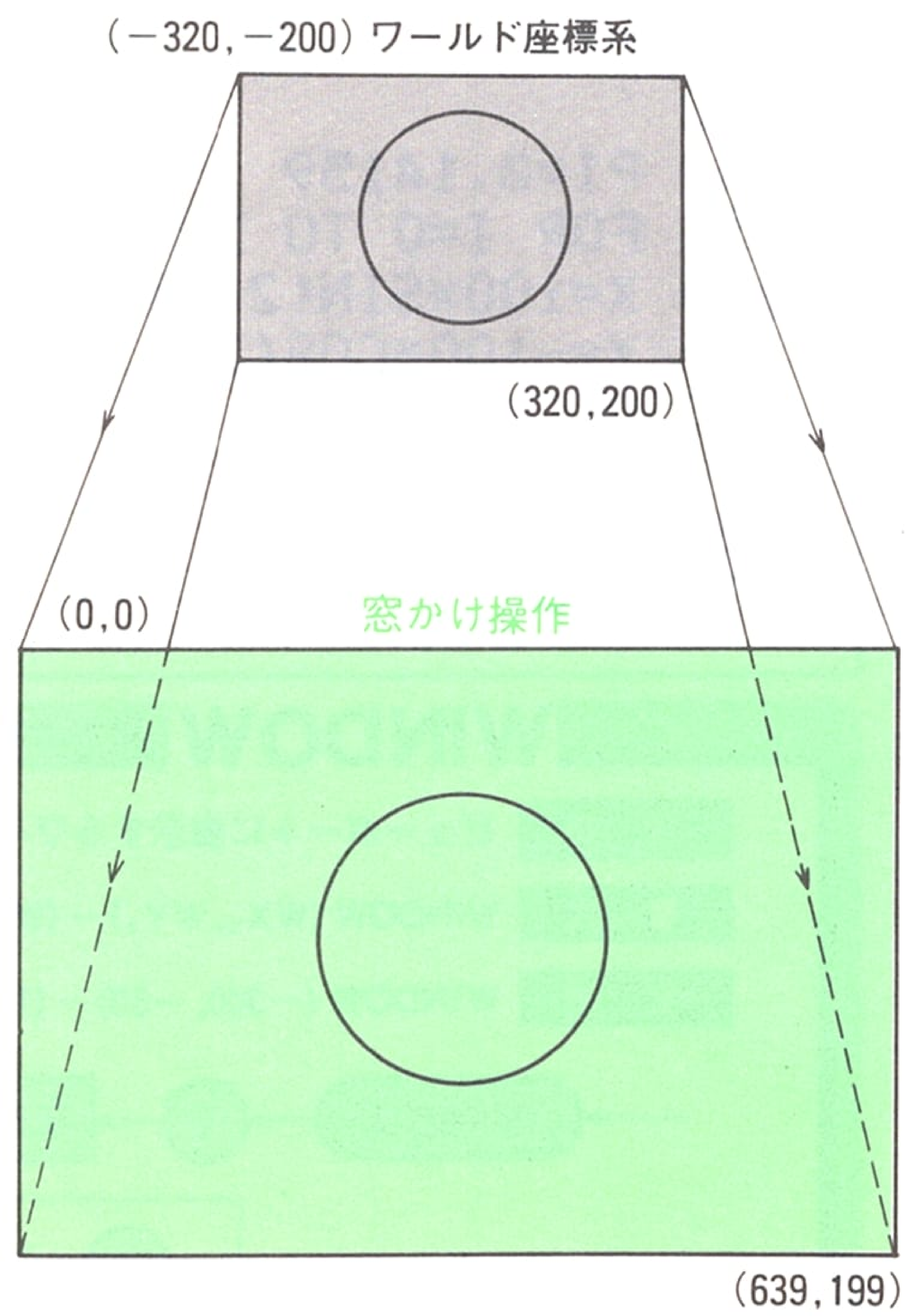
す。そこで比率を正しく守って、画面に表示させるには、窓のとり方にもう少し注意が必要だったのです。(−320, −200)と(320, 200) を端点に持つような長方形に選んでやりますと、うまく行きます。120行のWINDOW文を次のように変更してみました。

●プログラム 0−9●

```
100 CONSOLE 0,25,0,0
110 SCREEN 0,3:CLS 3:SCREEN 0,0
120 WINDOW (-320,-200)-(320,200)
130 '
140 PI=3.14159
150 FOR I=0 TO 200
160 X=100*SIN(2*PI*I/200)
170 Y=-100*COS(2*PI*I/200)
180 PSET(X,Y),4
190 NEXT I
200 '
210 END
```



●プログラム 0−8の場合●



●プログラム 0−9の場合●

ところで、これで画面上には、きちんとした円が描けるようになったのですが、プリンタに出すと、またまのびした縦長の楕円になってしまいます。画面に円を描く所まではやったのですから、ついでにプリンタに円が正しく出てくるように考えてみましょう。

☞ 1（P.203）

COPY 1～3までのコマンドを使ったのでは仕上がりがきわめて悪いのでCOPY 4を使おうと思います。COPY 4を使った時、縦横比率は1：1.3ですから、これを調整してやれば良いのです。

私も間違えたので、決してえらそうなことは申しませんが、ありうる失敗はWINDOW文の指定に次のようにしてしまうことです。

```
WINDOW (-320,-200*1.3)-(320,200*1.3)
```

これでは円はプリンタで出力すると、ペチャンコにつぶれてしまいます(プログラム 0―10)。

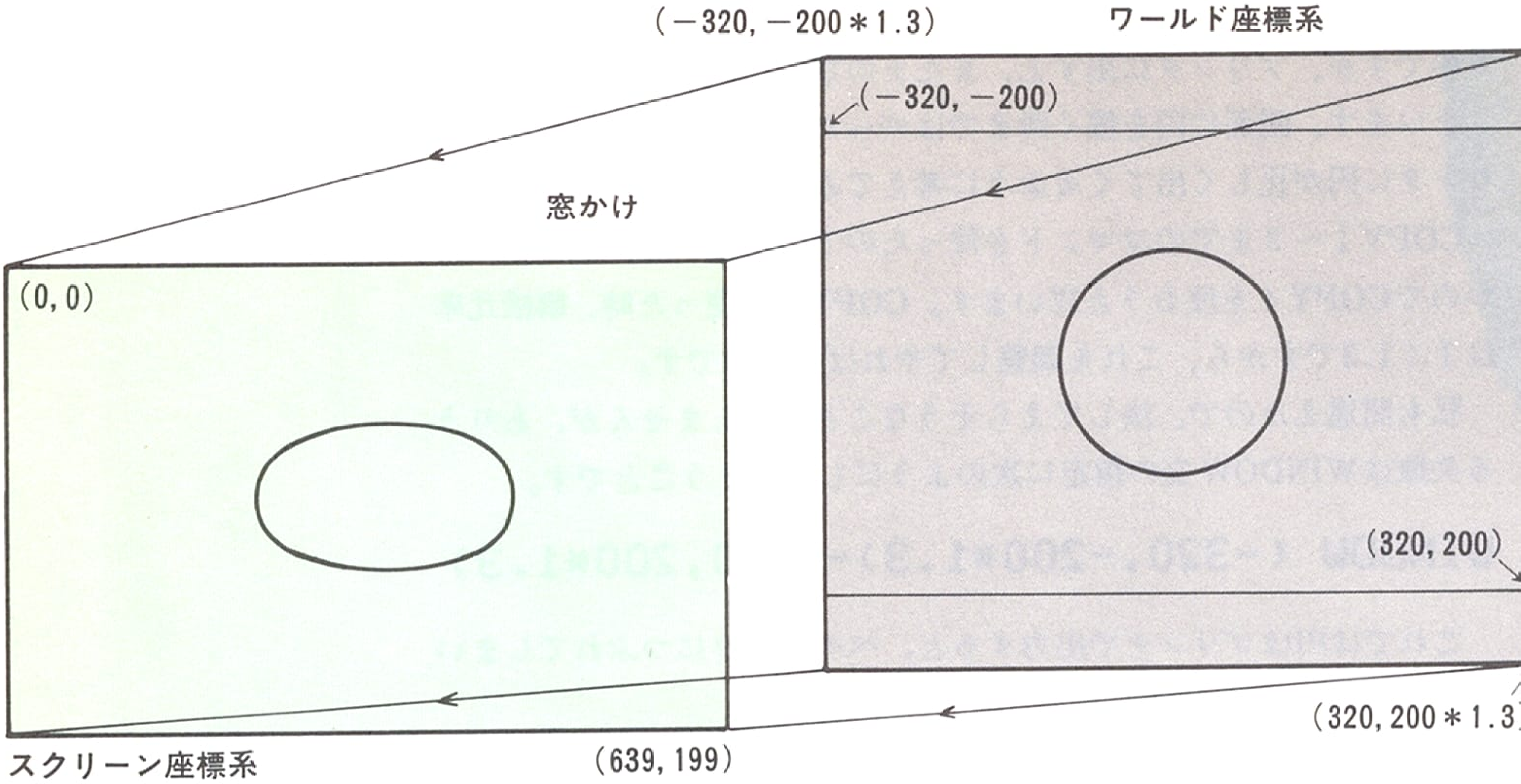
●プログラム 0―10●

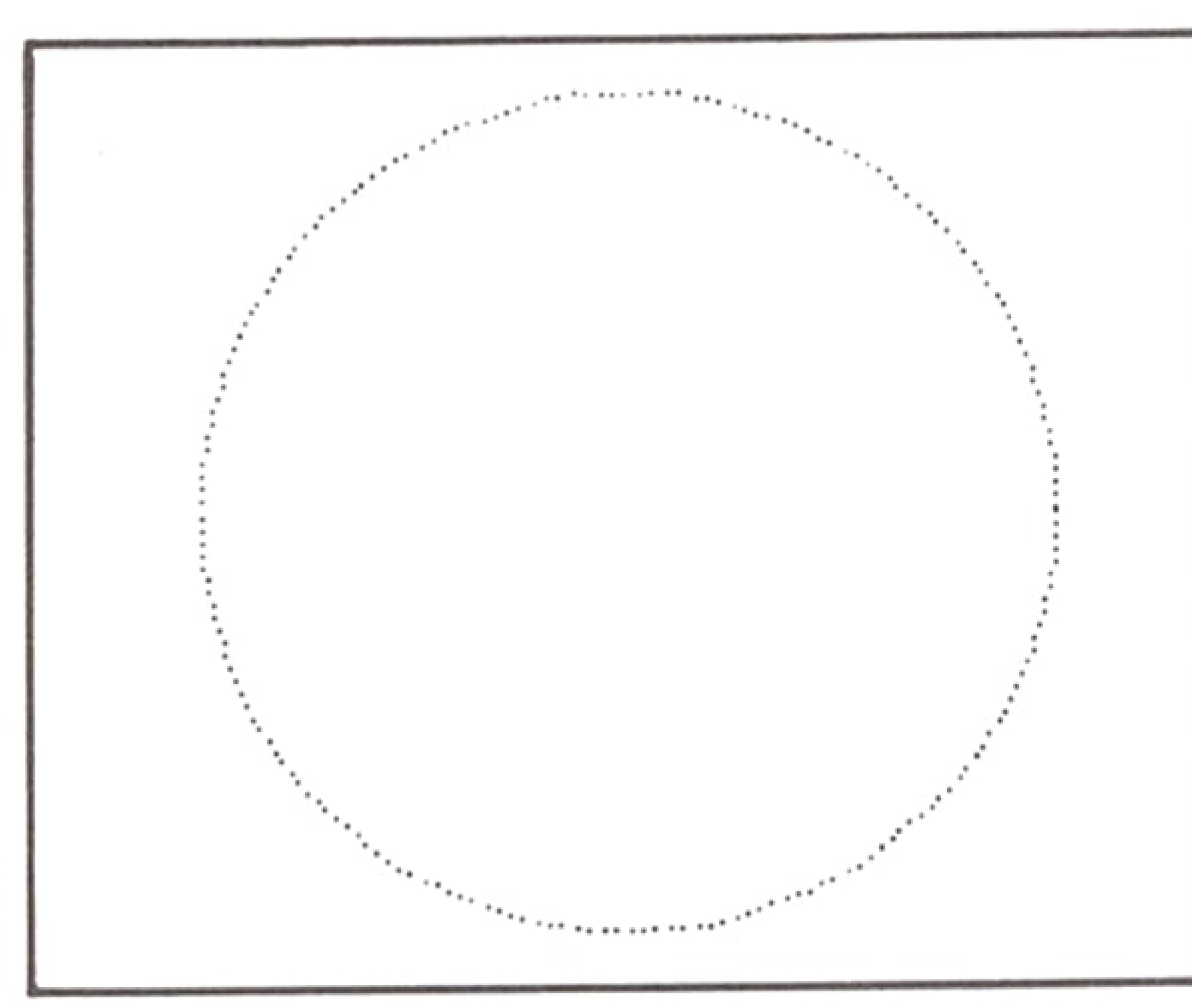
```
100 CONSOLE 0,25,0,0
110 SCREEN 0,3:CLS 3:SCREEN 0,0
120 WINDOW (-320,-200*1.3)-(320,200*1.3)
130 '
140 PI=3.14159
150 FOR I=0 TO 200
160 X=100*SIN(2*PI*I/200)
170 Y=-100*COS(2*PI*I/200)
180 PSET(X,Y),4
190 NEXT I
200 '
210 END
```

画面においては縦長になりますが、プリンタで正しく円を出力するためには次のようにWINDOW指定しなければなりません。

```
WINDOW (-320,-200/1.3)-(320,200/1.3)
```

プログラム0―11が修正したプログラムです。プリンタの出力例（次ページの図参照）をみると少しデコボコにみえますが、だいたいまともな円になっています。結構難しいものですね。




プリンタ用紙
(プログラム0−11の出力結果)

●プログラム0−11●

```
100 CONSOLE 0,25,0,0
110 SCREEN 0,3:CLS 3:SCREEN 0,0
120 WINDOW (-320,-200/1.3)-(320,200/1.3)
130 '
140 PI=3.14159
150 FOR I=0 TO 200
160 X=100*SIN(2*PI*I/200)
170 Y=-100*COS(2*PI*I/200)
180 PSET(X,Y),4
190 NEXT I
200 '
210 END
```

ついでにVIEW文を使ってみましょう。これはスクリーン座標系の中でどこにWINDOW文できり出されたワールド座標系でのグラフィックスを表示するかを指定するものです。これをVIEW文を使わずに実行しようとすると平行移動と縮小・拡大の座標変換を伴なっているので大変ややこしくなります。BASICだけでもできないことはありませんが、頭が痛くなります。

☞ 2 (P.211)

ここでのプログラム0−12は、VIEW文によって同じ円をいろいろに表示してみたものです。私の子供は帽子だと主張するのですが、どうでしょうか。

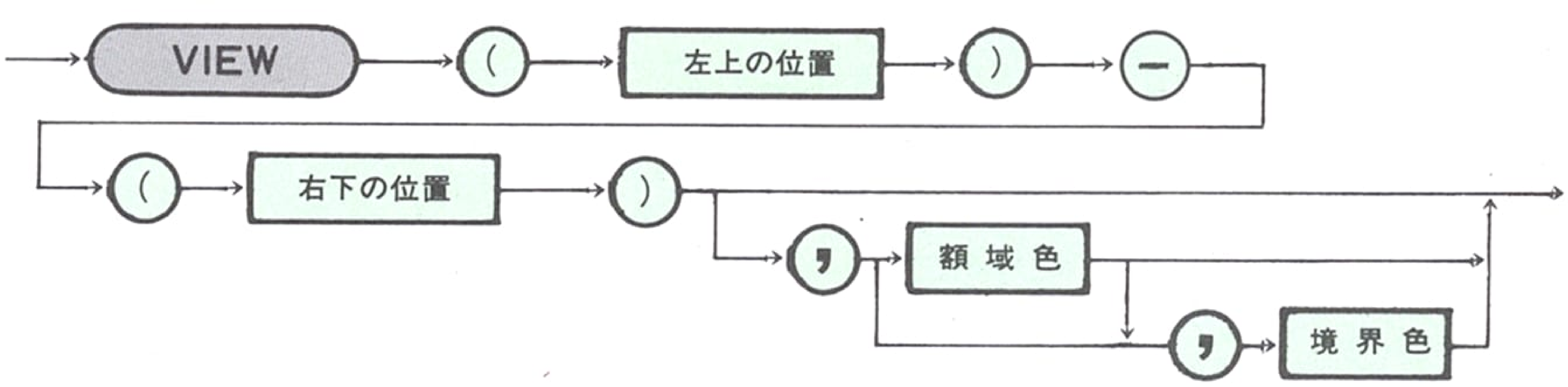
グラフィックスにとってWINDOWとVIEWは大切な役割を持ちます。これからの章では、くり返しくり返し登場してくることになると思います。

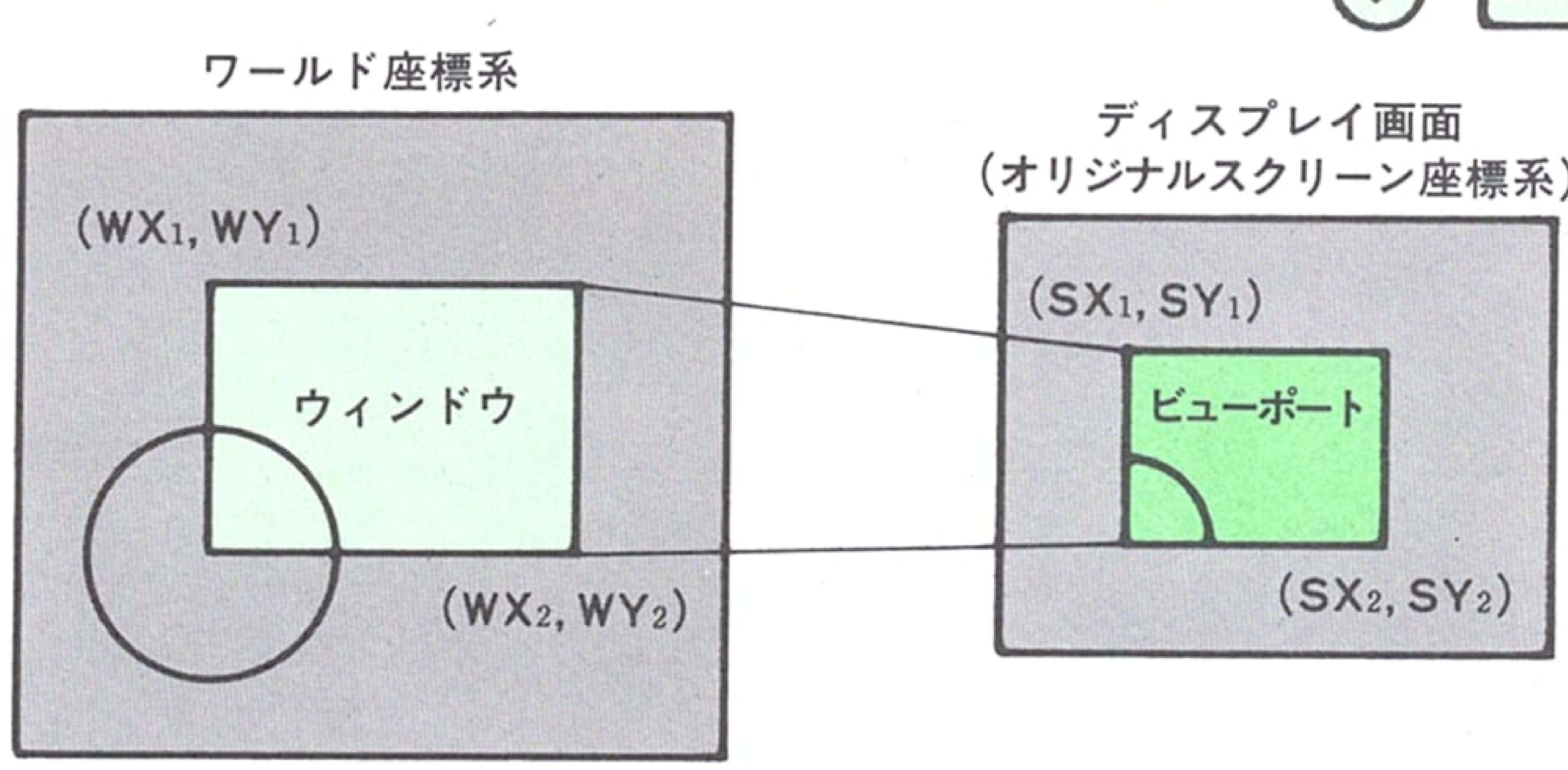
VIEW ●VIEW／ビュー

目的 ディスプレイ画面上での表示領域（ビューポート）を指定する

書式 VIEW (SX1, SY1)−(SX2, SY2)〔,〈領域色〉〕〔,〈境界色〉〕

文例 VIEW (100,30)−(200,75), ,7



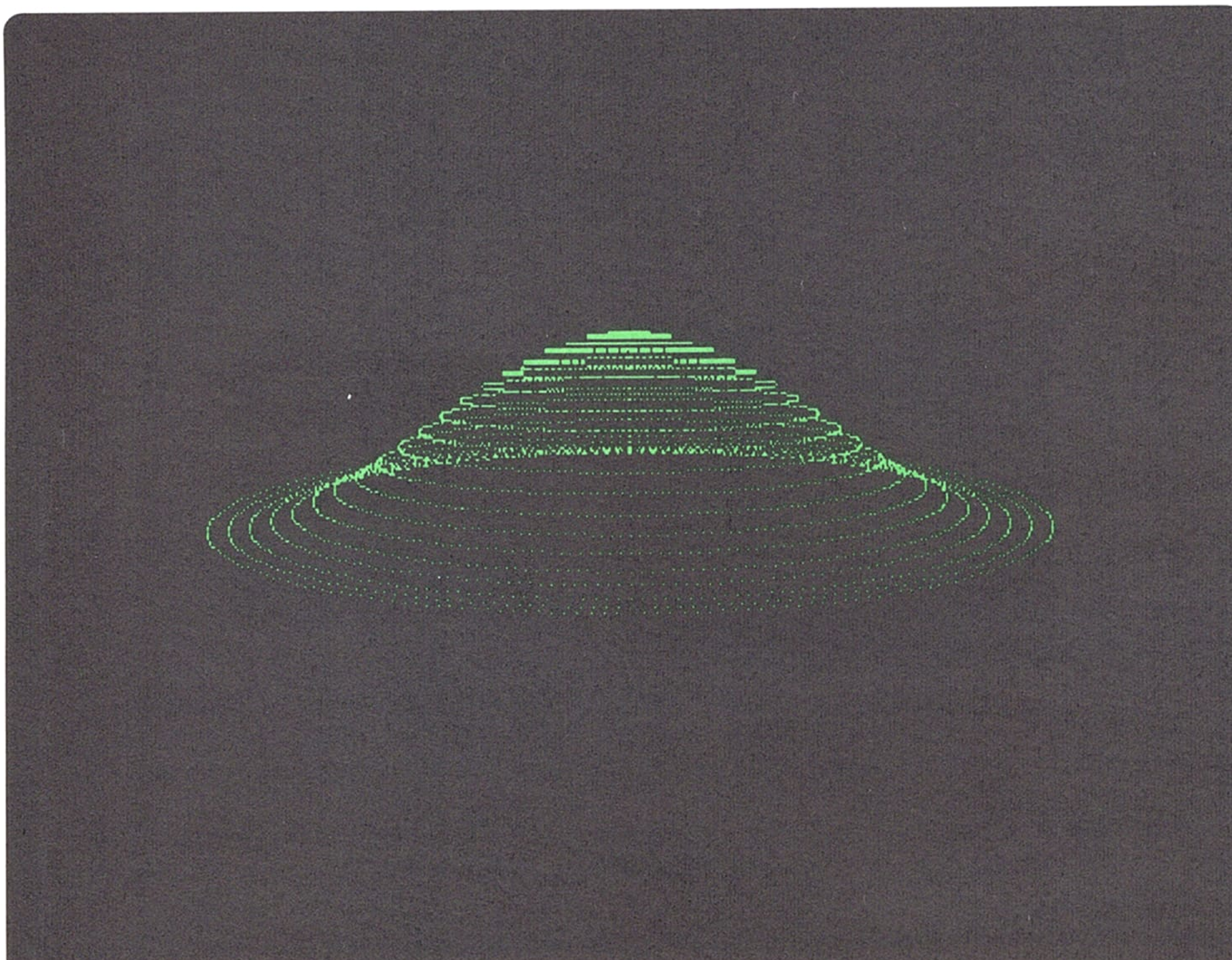


- VIEW文はWINDOW文によって指定されたワールド座標系上の領域内に入る図形を、ディスプレイ画面のどこに表示させるかというものです。
- 〈領域色〉が指定された場合には、指定されたパレット番号でビューポート内をぬりつぶします。
- 〈境界色〉が指定された場合には、指定されたパレット番号でビューポートのわくを描きます。CLSによって消去される画面の範囲は、このわくで囲まれた中だけで、わくは消されません。

●プログラム 0—12●

```
100 CONSOLE 0,25,0,0
110 SCREEN 0,3:CLS 3:SCREEN 0,0
120 WINDOW (-320,-480)-(320,480)
130 PI=3.14159
140 '
150 FOR I=0 TO 19
160 S=COS(2*PI*I/40)+1
170 VIEW (0,50)-(639,50+70*S)
180 FOR J=0 TO 200
190 X= 10*(20-I)*SIN(2*PI*J/200)
200 Y=-10*(20-I)*COS(2*PI*J/200)
210 PSET(X,Y),I MOD 7+1
220 NEXT J
230 NEXT I
240 '
```

●プログラムの実行結果



0・9　まとめ

これで第０章は終りです。これまで準備したことでPC-８８０１のグラフィックスは大変やりやすくなりました。CONSOLE, COLOR, SCREEN, WINDOW, VIEWなどのパラメータはかなり入り組んでいますから，一度にはとても覚えられないでしょう。一番良い勉強法は自分でいろいろプログラムを作ってみることです。本書の中でも特に説明なしに色々な技巧を使っている場合があります。そうした例にあたった場合、どうしてこういうやり方をしたのか考えてみてください。大切なことは面倒がらずにPC-８８０１のマニュアルを読み直すことですし、どういう工夫をしたら、もっとスマートにできるかを考えることです。実際こうしたやり方よりもっとよいやり方があると思いながらそのままにしてある所も沢山あります。アイデアが浮かんできたら、かならずPC-８８０１の前に座ってキーボードをたたいてください。頭の中で考えたことと実際は少し違うはずです。しかしインスピレーションには、どこか生かせる部分があるはずです。工夫をしてみてください。きっと上手なプログラムができると思います。友人と議論してみるのも良いことです。自分の知らない見方に気がつくでしょう。

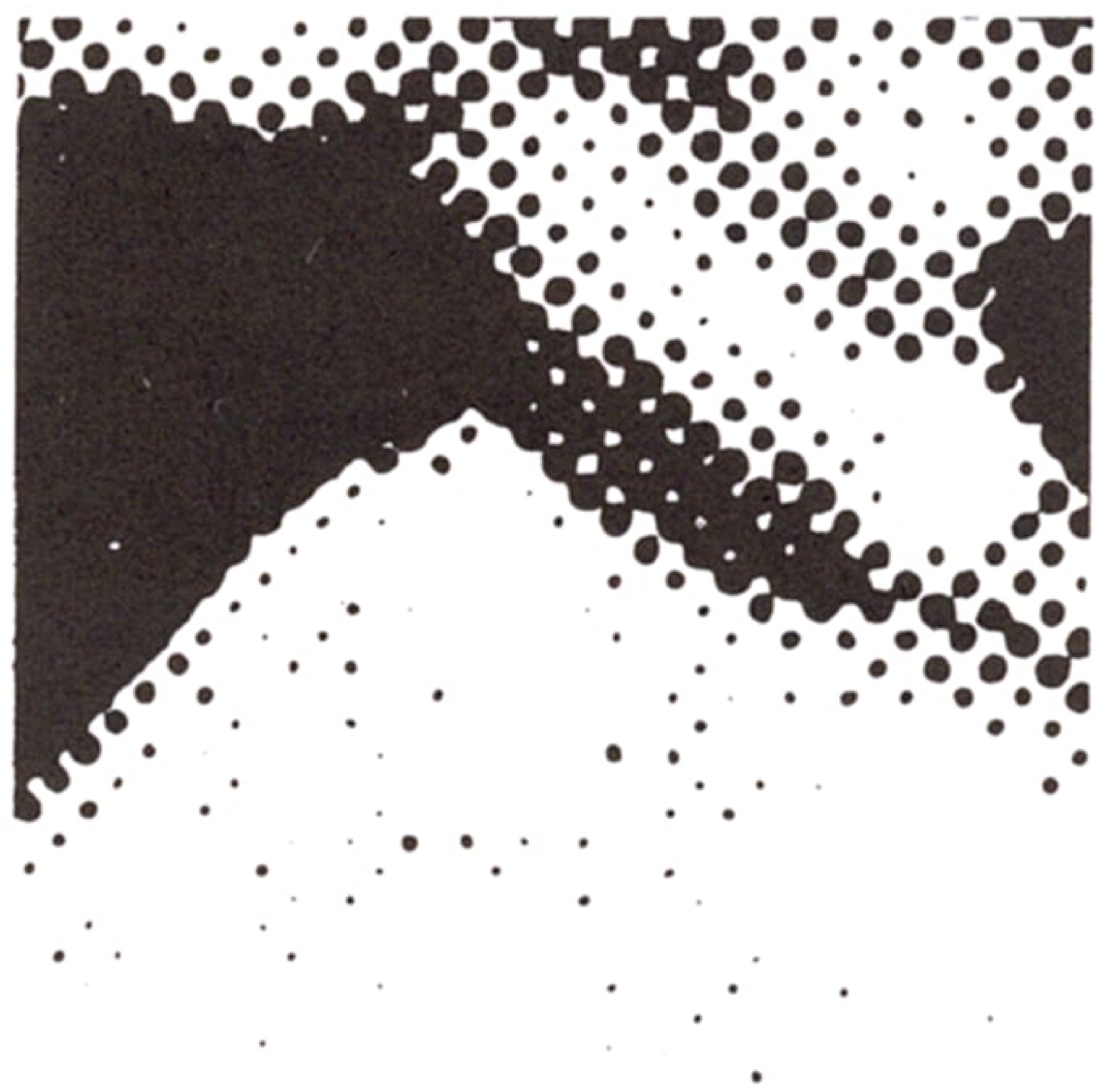
良いプログラムをみて、そこに使われているテクニックを分析し、ノートを作ることも大切です。自分で書くという行動は理解を深めます。

さらにグラフィックスを勉強するに当たって自分はなにを実現したいかという明確な目的意識を持つことも大切です。ただ漠然と勉強するというのでは読んだ本以上には進歩しません。いろいろな色彩を自由に使いこなしてファッションのデザインに応用してみたいとか、３次元グラフィックスで自動車の設計をしてみたいとか、建築デザインに使いたいとか具体的な目標を持ってください。そうすると、ただ漠然と勉強していた時よりコンピュータ・グラフィックスの問題点がはっきり見えてくるはずです。

欲を言えばアセンブリ言語の勉強をして頂けるともっと勉強が深くなると思います。特に大量のデータを扱うグラフィックスの場合アセンブリ言語が使えますと非常に高速にデータが処理できるようになります。たとえば3次元空間の中で物体をなめらかに回転させるとか、移動させる場合には欠くことのできないものとなります。全部をアセンブリ言語のプログラムで組むのは大変ですが、高速処理を必要とする部分に使うと効果があります。

第1章

簡単な図形を描く



1·0　相対座標と絶対座標

LP
(LAST REFERENCED POINT)

PC-８８０１のグラフィックスで難しいのは座標系のとり方だとお話してきました。ワールド座標系、スクリーン座標系、ビュー、ウィンドウなど新しい概念がぞくぞく入ってきています。そこにもう１つラストリファレンスト・ポイント(LP)という概念が入ってきます。ラストリファレンストというのは最後に参照したという意味です。最後と言っても現在の時点から見て、過去の一番最後ということで、日本語でいうと最も新しいということです。英語の語感は日本語の語感と少し違うのです。LAST WEEKというと『一番最後の』などという迷訳をしがちですが、これは『先週』と訳すのでしたね。

それでは何故ラストリファレンスト・ポイントLPが必要になるのでしょう。それは例題を通して説明することにしましょう。

いま、ここで画面に次々に点を表示していくことにします。点を表示するにはPSETを使います。いままでのやり方によればプログラム1-1のようにしました。

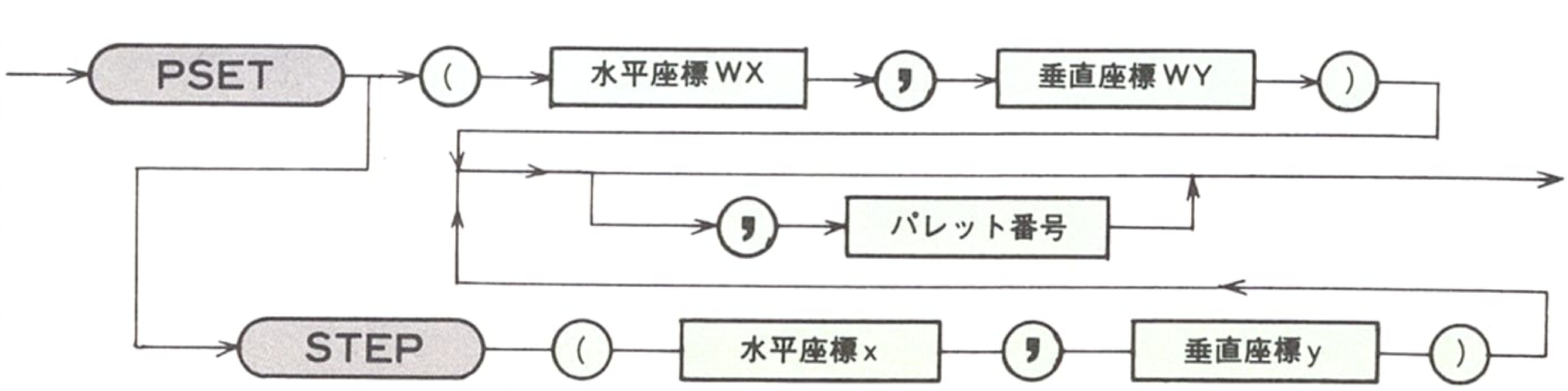
実際の場面でこういうアカ抜けしないプログラムを使うことは、とても考えられませんが、わかりやすいと思って作ってみました。行番号150から210までで、それぞれ指定された座標に色を変えて青から白の点を表示することになります。PSETはポイントセットの略だろうと思います。

絶対座標

プログラム1-1をもう少し短かくしたのがプログラム1-2です。こちらの方が短く楽になってはいます。この場合の座標の決定法は絶対座標というものを計算しながらすすむことになります。このやり方も良い方法ですが、座標を計算するのは絶対座標がすべてではありませ

PSET ●point set／ポイントセット

目　的	画面上の任意のドットをセットする
書　式	PSET (WX, WY) / STEP (x, y)〔, <パレット番号>〕
文　例	PSET (100,10), 4



●プログラム 1—1●

```
100 '  ***** PROGRAM 1-1 *****
110 '
120 SCREEN 0,3:CLS 3:SCREEN 0,0
130 WIDTH 80,25:CONSOLE 0,25,0,0
140 '
150 PSET(100,100),1
160 PSET(150,100),2
170 PSET(200,100),3
180 PSET(250,100),4
190 PSET(300,100),5
200 PSET(350,100),6
210 PSET(400,100),7
220 '
230 END
```

(150～210: 水平に7色の点を表示する)

●プログラム 1—2●

```
100 '  ***** PROGRAM 1-2 *****
110 '
120 SCREEN 0,3:CLS 3:SCREEN 0,0
130 WIDTH 80,25:CONSOLE 0,25,0,0
140 '
150 FOR I=1 TO 7
160 X=50+50*I
170 PSET(X,100),I
180 NEXT I
220 '
230 END
```

(150～180: 50ドットおきに7色の点を表示する)

ん。今の点のとり方は要するにX座標(横座標のことです)を50ずつ移動させているにすぎませんから、最後にかいた点から50だけ横に移動するという操作をくり返していっても良いわけです。何だ、かえって面倒そうだなと思われるかも知れませんが、決してそうではありません。

相対座標

●相対性理論は無関係ですよ

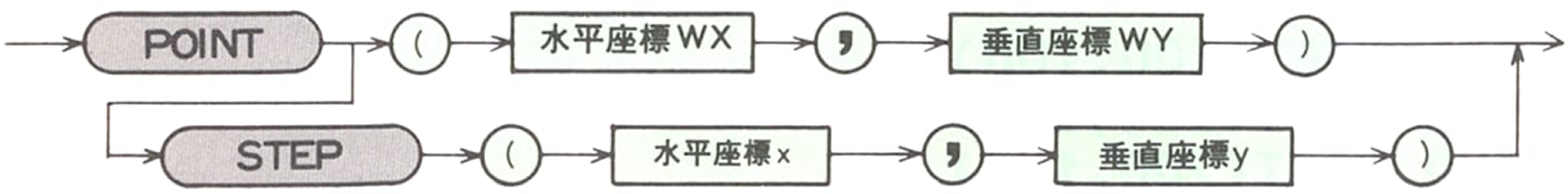
相対座標というものによる新しい考え方のプログラム1—3を見て下さい。プログラムの長さ自体は同じですし、横へ50だけ動くということを使っています。行番号150に登場するPOINTがラストリファレンスト・ポイントLPを設定しています。このプログラムの場合は始めにどこに点をとるかを明示しています。相対座標という考え方は前の点から見て相対的に次の点がどこにあるかを決める方式ですから、最初の点の位置だけは明確にしておかなくてはならないのです。PC-８８０

POINT ●point／ポイント

目　的　ラストレファレンスト・ポイント(LP)を変更する

書　式　POINT |(WX, WY)|
　　　　　　　　|STEP (x, y)|

文　例　POINT (－200, 30)



●プログラム 1―3●

```
100 '  ***** PROGRAM 1-3 *****
110 '
120 SCREEN 0,3:CLS 3:SCREEN 0,0
130 WIDTH 80,25:CONSOLE 0,25,0,0
140 '
150 POINT (50,100)
160 FOR I=1 TO 7
170 PSET STEP(50,0),I
180 NEXT I
220 '
230 END
```

150行：LPを座標(50,100)に設定する

170行：すぐ前にセットされた点からX座標を50ドットずれたところへ新しい点をセットする

1の場合、POINT文を使わないでLPを明示しないと、LPは(0,0)に指定されるようです。

なぜ相対座標が良いかといいますと、プログラムを少し変更して図形の表示位置を変更させる場合、POINT文だけをいじれば、あとは何の変更も必要もないからです。絶対座標ですと座標系の複雑な変更を必要とする場合があります。

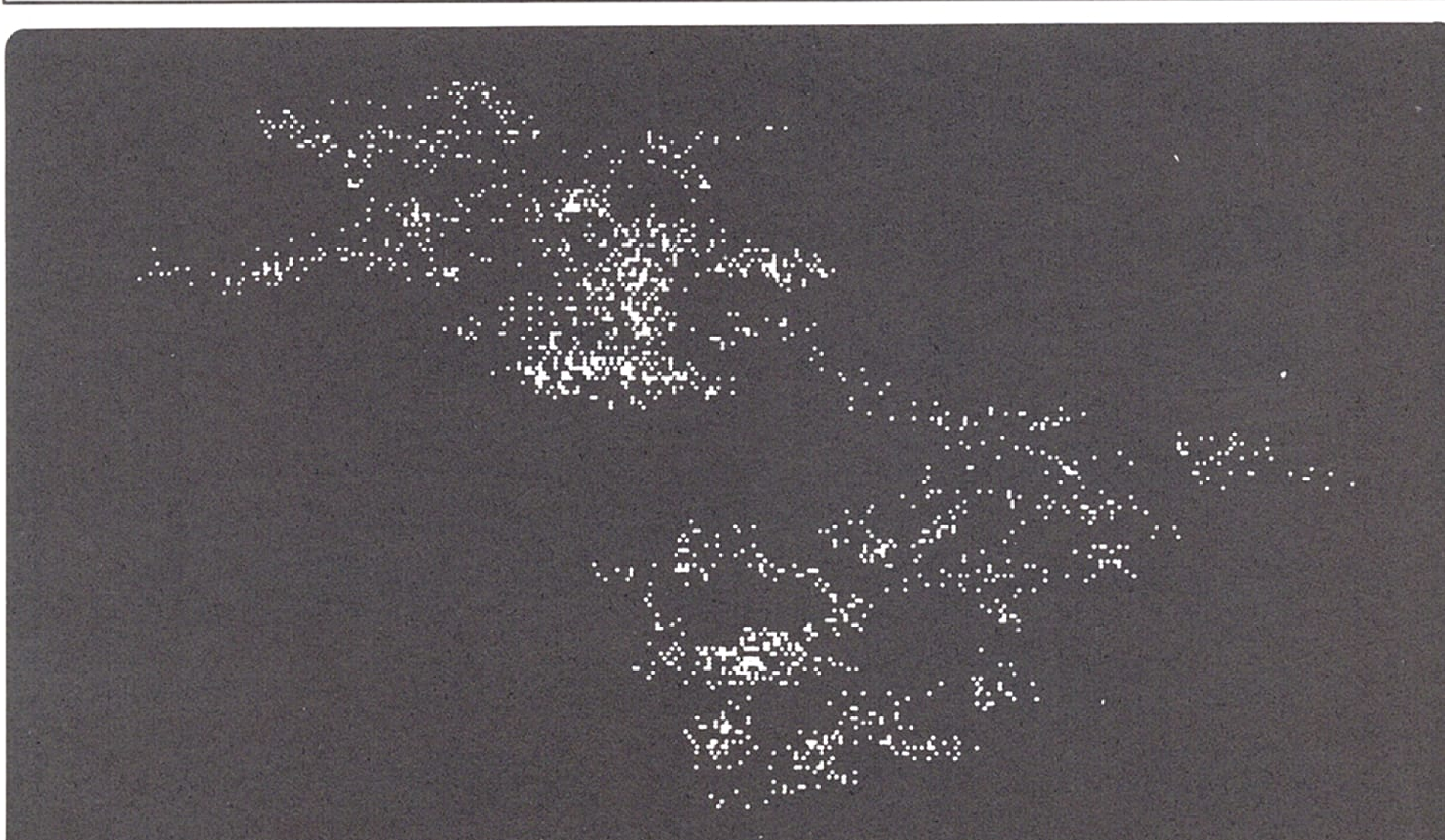
アセンブリ言語というものでプログラムを組んだ方は良くご存知のことと思いますが、絶対番地方式で組んだプログラムは少しでも番地を変えて組むと全面的に番地を変更してやらなければならないというわずらわしさがあるのですが、相対番地方式ですと、プログラムをメモリーの中で移動させてもあまり変更を必要としないので大変便利なのです。

相対座標を使ったプログラムをいくつか使ってみましょう。よく、PSETでRND関数を使った例をみます。ここでは点の動ける距離をある範囲で限定して、点がランダム運動するようすを見ることにしましょう。次のプログラムを入力して走らせてみて下さい。

点が7色に色を変えながらアメーバのように動きまわり始めるでしょう。RND関数には必ずくせがあるので、画面全体を均等に動きまわることはありません。このプログラムではWINDOW文を使って調節し

●プログラム 1-4●

```
100 ' ***** PROGRAM 1-4 *****
110 '
120 SCREEN 0,3:CLS 3:SCREEN 0,0
130 WIDTH 80,25:CONSOLE 0,25,0,0
140 WINDOW (-320,-200)-(320,200)
150 '
160 P=3.14159
170 POINT (0,0)          ←ワールド座標上の(0,0)にLPを設定
180 '
190 FOR I=0 TO 2000      ┐
200 X=16*(RND(1)-.5)     │
210 Y=10*(RND(1)-.5)     ├ドットをランダムにセットする
220 PSET STEP(X,Y),I MOD 8 │
230 NEXT I               ┘
240 '
250 END
```



ています。絶対座標を使った時とは違った動き方をするのでおもしろさが増すでしょう。

線を引くのにはLINE文を使います。ふつうの絶対座標による使い方ではLINE文の構造図にもあるように、

```
LINE (0,0)-(639,199),4
```

とすればよいのです。座標はワールド座標で指定できますが、特に意識しなくともかまいません。

PSET文の場合と同様にLINE文でもSTEPを使うことができます。始点に対して終点を示す絶対座標の場合とは違って、STEPを使う時には相対座標ですから、始点に対して相対的にどれだけ動いたかをSTEP量で示すことになります。たとえば、

```
LINE (100,100)-STEP(100,100),4
```

とあれば始点が(100,100),終点は(200,200))にある線分を緑色で書くことになります。もう少し大胆な使い方ですと、

```
LINE -STEP(100,100),4
```

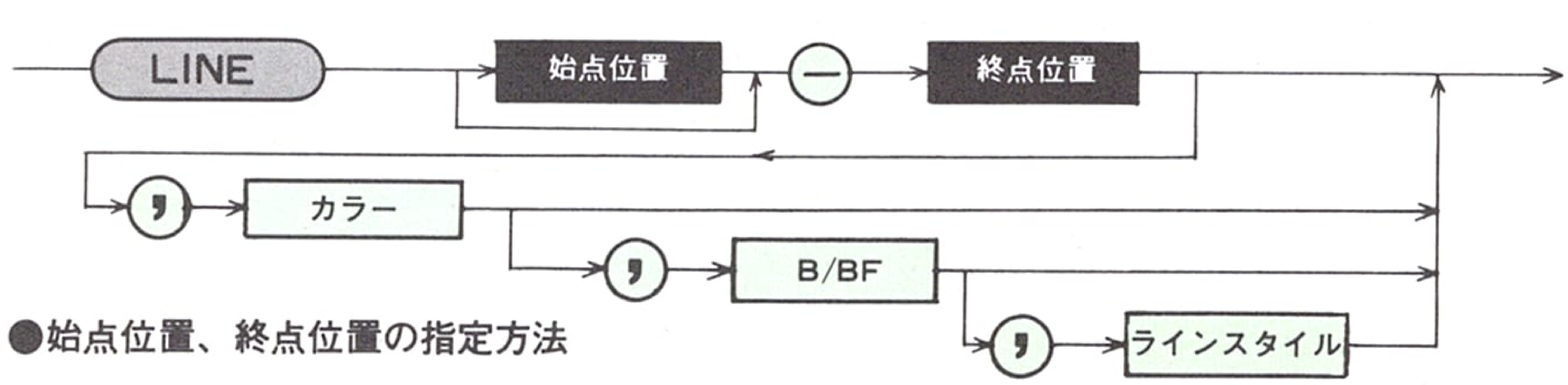
というのがあります。始点はLP(ラストリファレンスト・ポイント)になります。この形式を使う場合には線分の引き始めの座標をPOINT文でおさえておく必要があります。この形式のLINE文を使ったのがプ

LINE ●line／ライン

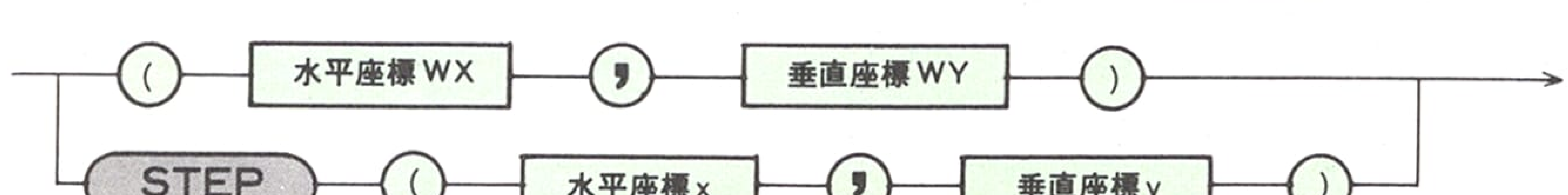
目的 線を描く

書式
```
LINE〔|     (WX, WY)|〕－|     (x, y)|
     |STEP (WX, WY)|    |STEP (x, y)|
    〔,〈パレット番号〉〕〔,|B |〕〔,〈ラインスタイル〉〕
                         |BF|
```

文例 LINE (100,100) − (135,100) ,7, ,&HF99F



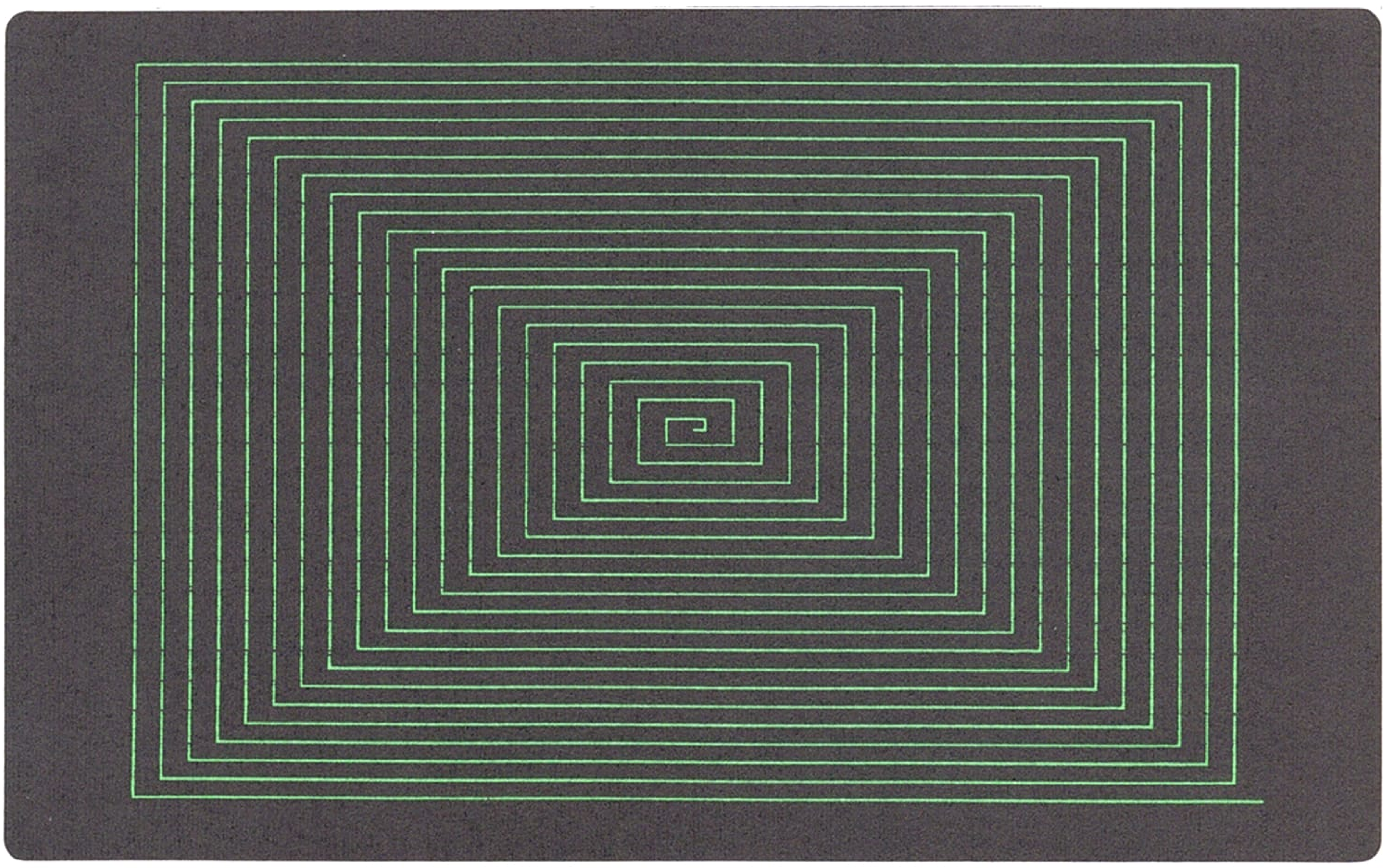
●始点位置、終点位置の指定方法



ログラム1―5です。うずまきになって色をかえながら、グルグルと回って行きます。なかなか面白いですよ。相対座標のSTEP量をきめる方法はもっと、こった方法もたくさんあります。みなさんも考えてみて下さい。

●プログラム 1―5●

```
100 ' ***** PROGRAM 1-5 *****
110 '
120 SCREEN 0,3:CLS 3:SCREEN 0,0
130 WIDTH 80,25:CONSOLE 0,25,0,0
140 WINDOW (-160,-100)-(160,100)
150 '
170 POINT (0,0)
180 '
190 FOR I=0 TO 80
200 N=(I MOD 4)+1
210 ON N GOTO 220,230,240,250
220 LINE -STEP(5*(I+1)/2,0),N:GOTO 260 ←右へ線を引く
230 LINE -STEP(0,5*(-I/2)),N:GOTO 260 ←上へ線を引く
240 LINE -STEP(-5*(I+1)/2,0),N:GOTO 260 ←左へ線を引く
250 LINE -STEP(0,5*I/2),N:GOTO 260 ←下へ線を引く
260 NEXT I
270 '
280 END
```



●実際の画面は横長でなく縦長になります

PRESET

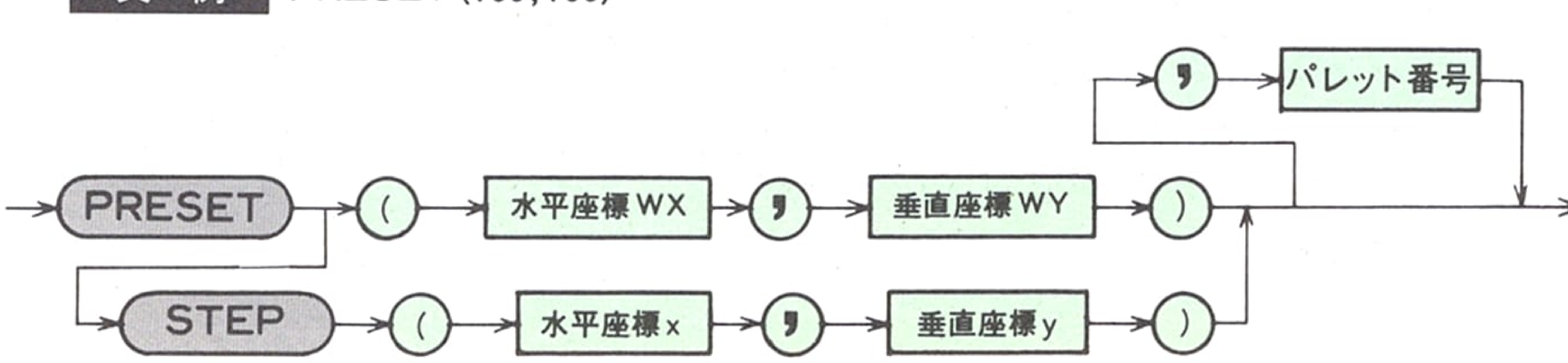
PSETを勉強したのですからPRESET(ポイント・リセット)文も勉強しておきましょう。PRESETはバックグランド・カラーと同じ色で点を表示します。結局点は見えなくなる効果があります。 この場合も絶対座標による表示の他に相対座標による表示もできます。プログラム1―6はPSETとPRESETを使った例です。この例では7色の点が放物線の弧をかきながら、次々に表示されたあと、画面の右端にたどりつくと、左側から放物線の弧が消えて行きます。これがくり返されると放物線の弧が動いていくようにに見えます。

●プログラム 1―6●

```
100 ' ***** PROGRAM 1-6 *****
110 '
120 SCREEN 0,3:CLS 3:SCREEN 0,0
130 WIDTH 80,25:CONSOLE 0,25,0,0
140 '
150 FOR I=1 TO 20          ┐
160 POINT(30*I,100)        │
170 FOR J=1 TO 7           ├─放物線の弧を描く
180 PSET STEP(3,J),J       │
190 NEXT J                 │
200 NEXT I                 ┘
210 '
220 FOR I=1 TO 20          ┐
230 POINT(30*I,100)        │
240 FOR J=1 TO 7           ├─表示されている放物線を次々と消す
250 PRESET STEP(3,J)       │
260 NEXT J                 │
270 NEXT I                 ┘
280 '
290 GOTO 150
```

PRESET ●point reset／ポイントリセット

目　的	画面上の任意のドットをリセットする
書　式	PRESET |(WX, WY) / STEP (x, y)| 〔,〈パレット番号〉〕
文　例	PRESET (100,100)



1・1 円はどうやって描くのだろう

さて円を描く例題に入ることにしましょう。構造図を見るとゴチャゴチャ一杯書いてありますね。これだけの機能が盛り込まれているわけです。全部は使わなくとも良いのです。円を描くのに必要なのは中心と半径だけですから、次のようなプログラムでも円は十分描けます。

●プログラム 1−7●

```
100 '***** PROGRAM 1-7 *****
110 '
120 SCREEN 0,3:CLS 3:SCREEN 0,0
130 WIDTH 80,25:CONSOLE 0,25,0,0
140 WINDOW (-320,-100)-(320,100)
150 CIRCLE (0,0),100
160 END
```

よく間違えるのは行番号140のWINDOW文を忘れてしまうことです。WINDOW文を忘れると画面の左上すみに4分の1円が表示されるにすぎなくなります。 プログラム 1-8を実行してみて下さい。

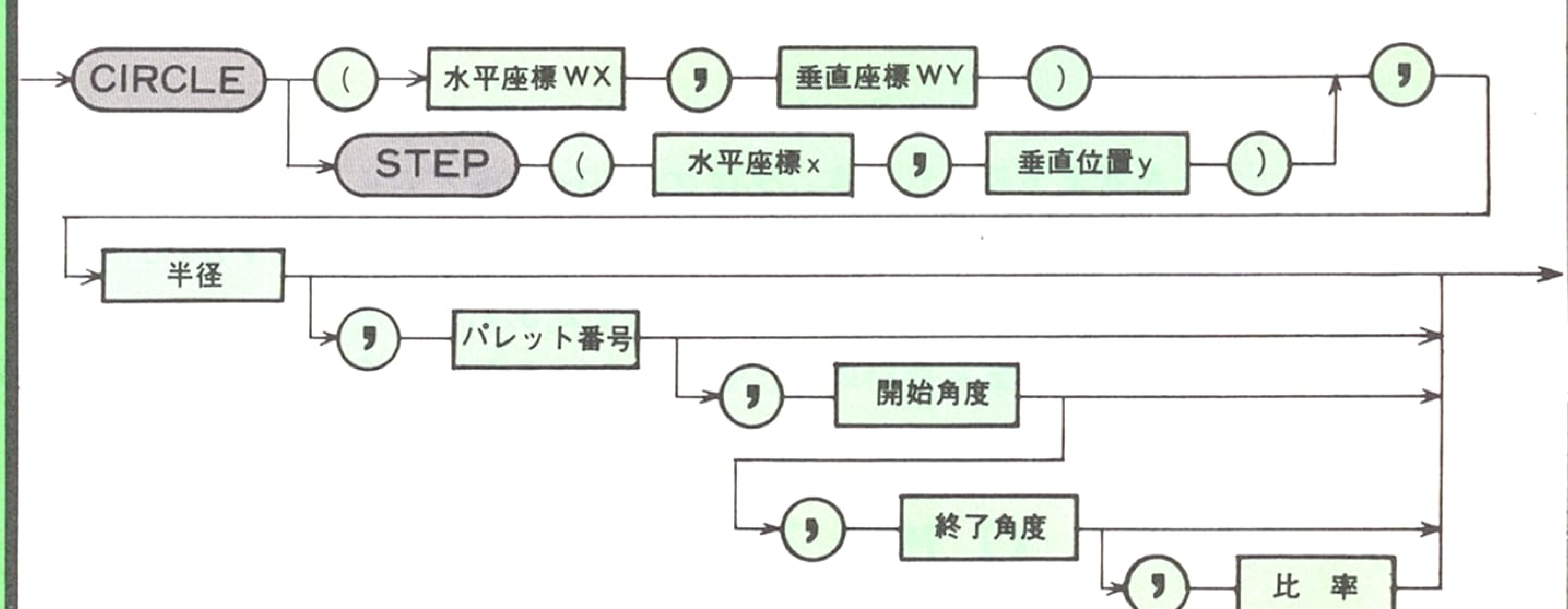
WINDOW文を使うのがいやだとなると、座標変換してやる他にありません。プログラム1−9がその例です。

CIRCLE ●circle／サークル

目的 円(楕円)および弧の作図

書式 CIRCLE {(WX, WY) | STEP (x, y)} ,<半径>〔,<パレット番号>〕〔,<開始角度>〕〔,<終了角度>〕〔,<比率>〕

文例 CIRCLE (320,100),100,2



●プログラム 1−8●

```
100 '***** PROGRAM 1-8 *****
110 '
120 SCREEN 0,3:CLS 3:SCREEN 0,0
130 WIDTH 80,25:CONSOLE 0,25,0,0
140 '
150 CIRCLE (0,0),100   ←円の第４象限のみが画面に表示される
160 END
```

●プログラム 1−9●

```
100 '***** PROGRAM 1-9 *****
110 '
120 SCREEN 0,3:CLS 3:SCREEN 0,0
130 WIDTH 80,25:CONSOLE 0,25,0,0
140 '
150 CIRCLE(320,100),100   ←中心の座標を移動すれば円の
                            全体が画面に点示される
160 END
```

●カラーコードの覚え方は22ページにあります。暗記してしまいましょう。

やはり円には色をつけたいですね。パレット番号のオプションをつけてやれば色がつきます。赤の円を描いてみましょう。プログラム1−7、プログラム1−9の行番号150をそれぞれ次のように変更して実行してみて下さい。

```
150 CIRCLE (0,0),100,2

150 CIRCLE (320,100),100,2
```

それでは色を変えて同心円を描きたいという注文にはどうこたえたら良いでしょう。少し考えてみて下さい。

考えましたか？　考えないと力はつきませんよ。それでは一緒に答を考えてみましょう。プログラム1−10をごらん下さい。

MOD

7色の作り方にMODを使います。御存知と思いますが、I　MOD　7の計算の結果、出てくる答は0,1,2,3,4,5,6　です。

黒はつまらないので、0がでないように1だけシフトしてやりたいですね。そこで、

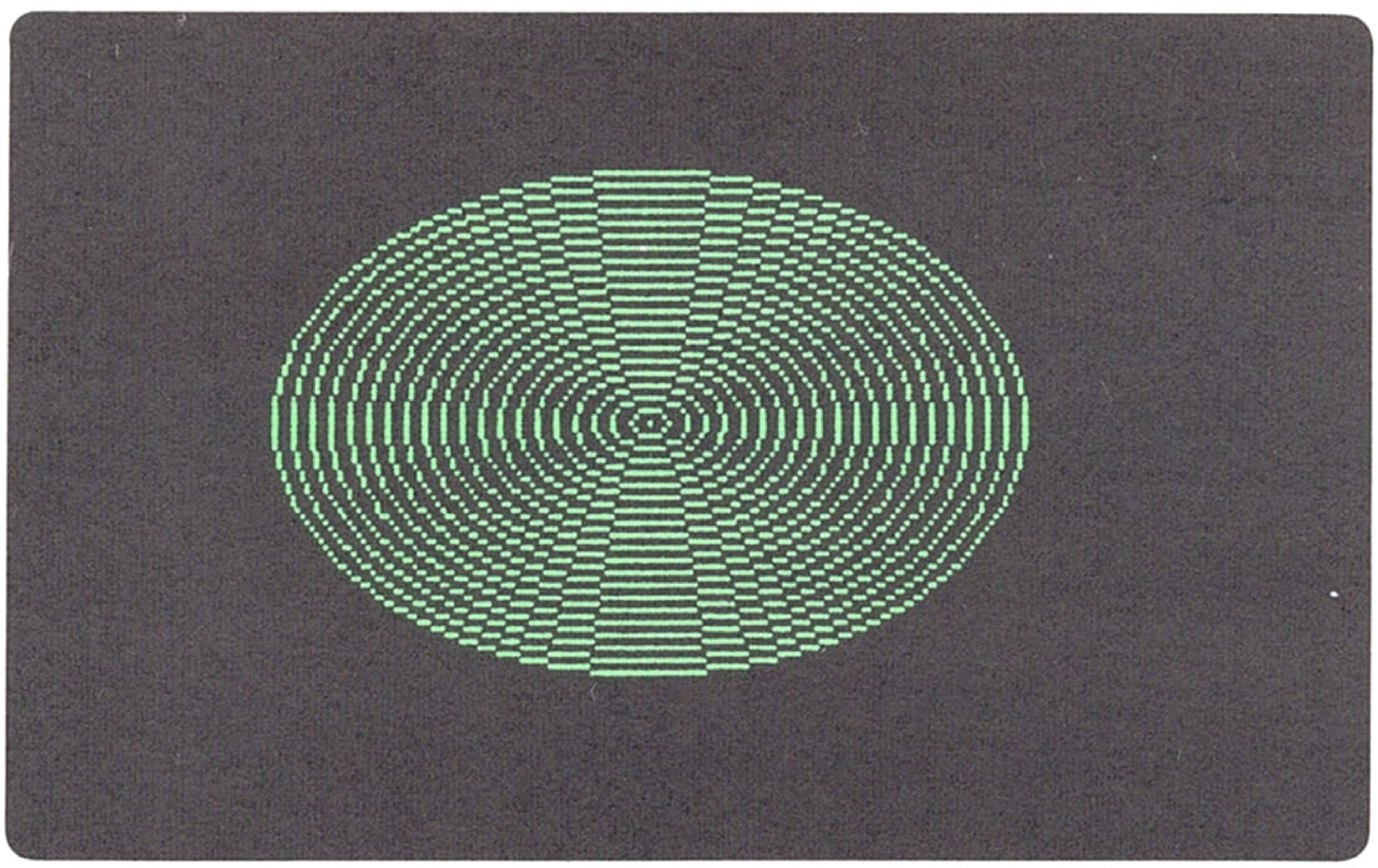
```
(I MOD 7)+1
```

として、1,2,3,4,5,6,7がでるよう工夫しているのです。I MOD 7はIを7で割った余りとも考えられます。

●プログラム 1−10●

```
100 '***** PROGRAM1-10*****
110 '
120 SCREEN 0,3:CLS 3:SCREEN 0,0
130 WIDTH 80,25:CONSOLE 0,25,0,0
140 WINDOW (-320,-100)-(320,100)
150 '
160 FOR I=0 TO 20
170 CIRCLE (0,0),I*4,(I MOD 7)+1
180 NEXT I
190 '
200 END
```

青、赤、紫、緑、水色、黄、白の円が内側から順番に描かれていく



次に円が中心を移動させながら動いていくものを考えましょう。

160行のPOINT文を忘れないようにして下さい。7色の円が横に一列に並びます。

●プログラム 1−11●

```
100 '***** PROGRAM1-11*****
110 '
120 SCREEN 0,3:CLS 3:SCREEN 0,0
130 WIDTH 80,25:CONSOLE 0,25,0,0
140 WINDOW (-320,-100)-(320,100)
150 '
160 POINT (-220,0)
170 FOR I=1 TO 7
180 CIRCLE STEP(60,0),20,(I MOD 7)+1
190 NEXT I
200 '
210 END
```

160: LPの設定

180: 中心を移動させて７つの円を描く

1·2 「手まり」を回してみる

これまでいろいろなテクニックを勉強してきました。ここらあたりで総合的なテクニックを使って、面白い例題を作ってみましょう。最近はほとんど見かけませんが、手まりというのが昔ありました。これに色を塗ってまわすことを考えてみましょう。とは言っても、手まりを完全に実現するのは難しいのでイメージ的に似ていれば、それで満足することにします。

まず手まりには赤と紫と緑の糸を使うことにします。3色にするのは手まりをまわすのに都合が良いからです。手まりの表面を色わけするには手まりの表面を分割してやらねばなりません。

プログラムを下図に示します。行番号180から行番号220の間で放射状の線をひいています。また行番号240から行番号260では同心円をひいていますが、手まりに立体感がでるように半径は三角関数を使って変化させています。行番号280から行番号290は手まりを浮き上が

●プログラム 1—12●

```
100 '***** PROGRAM1-12*****
110 '
120 SCREEN 0,3:CLS 3:SCREEN 0,0
130 WIDTH 80,25:CONSOLE 0,25,0,0
140 WINDOW (-160,-100)-(160,100)
150 '
160 PI=3.14159
170 '
180 FOR I=0 TO 360 STEP 20
190 X=100*COS(I*PI/180)
200 Y=100*SIN(I*PI/180)
210 LINE (0,0)-(X,Y),6
220 NEXT I
230 '
240 FOR I=0 TO 9
250 CIRCLE (0,0),100*SIN(PI*I/18),6
260 NEXT I
270 '
280 PAINT (-160,-100),1,6
290 PAINT (160,100),1,6
300 '
310 END
```

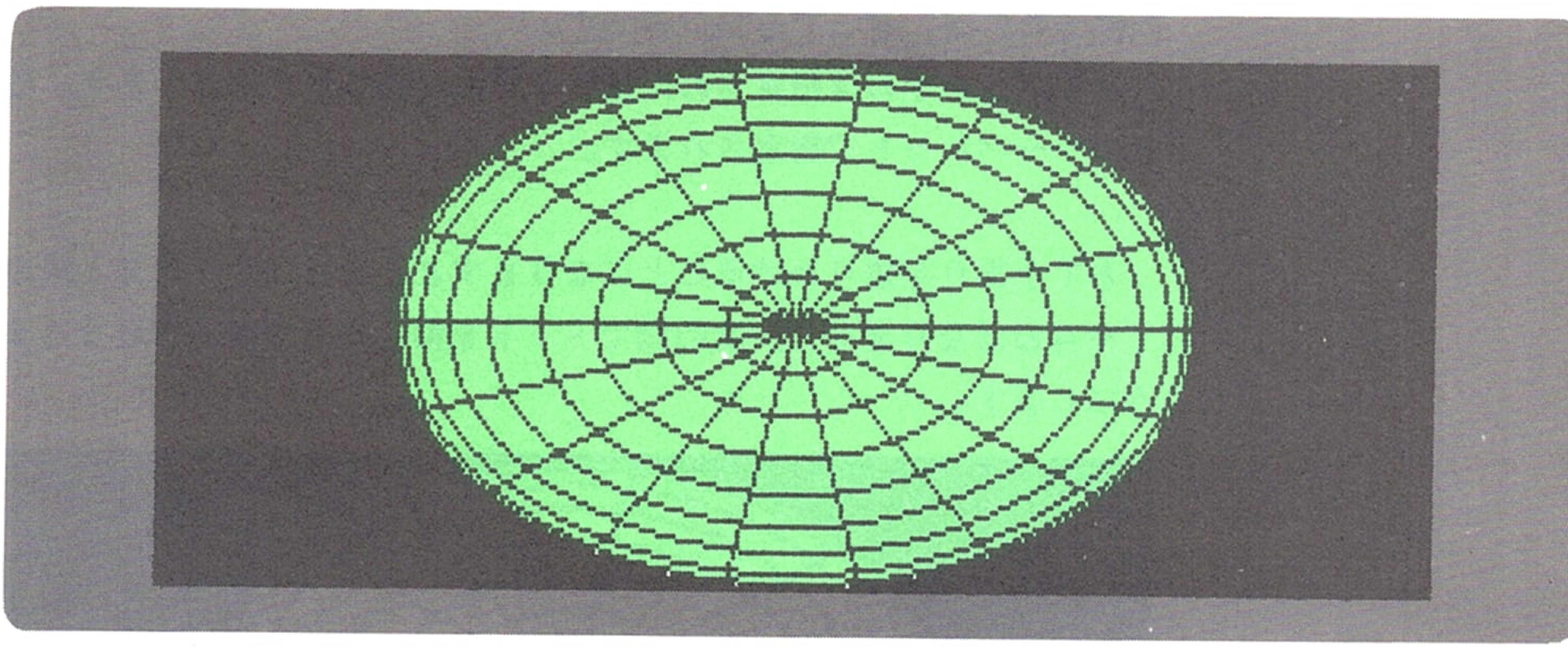
180〜220：原点から18本の放射状の線をひく

240〜260：同心円を描く

280〜290：背景を青く塗る

らせるように背景を青で塗っています。

プログラム 1—12の実行結果を次に示します。楕円になっていますが実際は正円です。感じをつかんでいただけばよいと思います。



行番号280、290、にPAINTというのがありますが、これは絵具を定められた領域に塗ってくれる命令です。構文図を見れば使い方がわかると思いますが、簡単に説明しておきます。

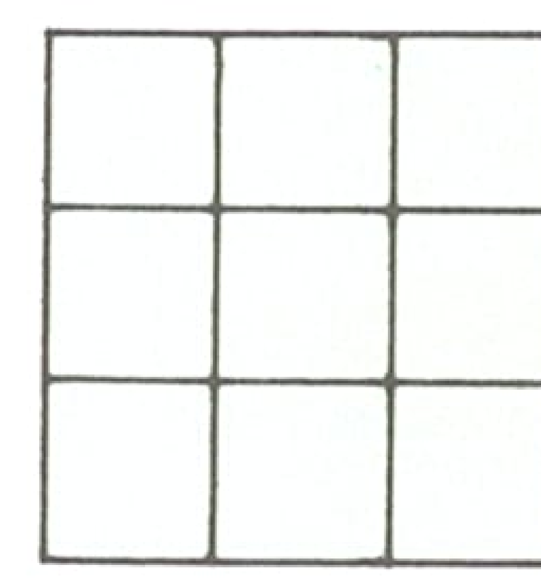


図1

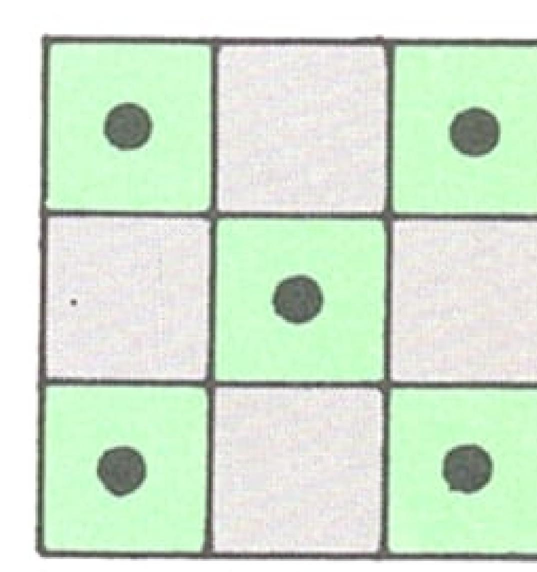


図2

色を塗る場合には、どこからどこまで何色に塗るかが大切な問題となります。どこからどこまでを指定するのがバウンダリーカラーという境界色です。PC-８８０１では境界色は１色しか指定できないので注意する必要があります。これがFMシリーズでは境界色は何色でも指定できました。

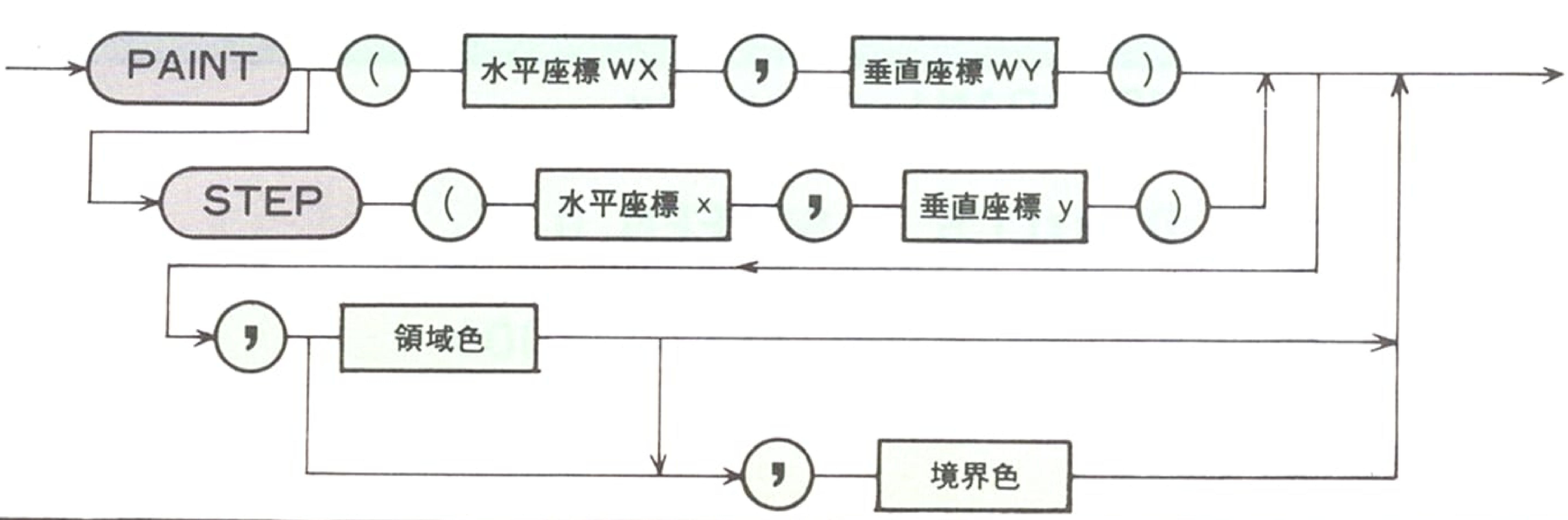
『どこからどこまで』という中には、境界色だけでは指示できない部分があります。たとえば右図のような格子を考えてみましょう。『黄色

PAINT ●paint／ペイント

目的 指定された境界色で囲まれた領域を指定された色で塗る

書式 PAINT ｜ (WX, WY) / STEP(x,y) ｜ 〔,〈領域色〉〔,〈境界色〉〕

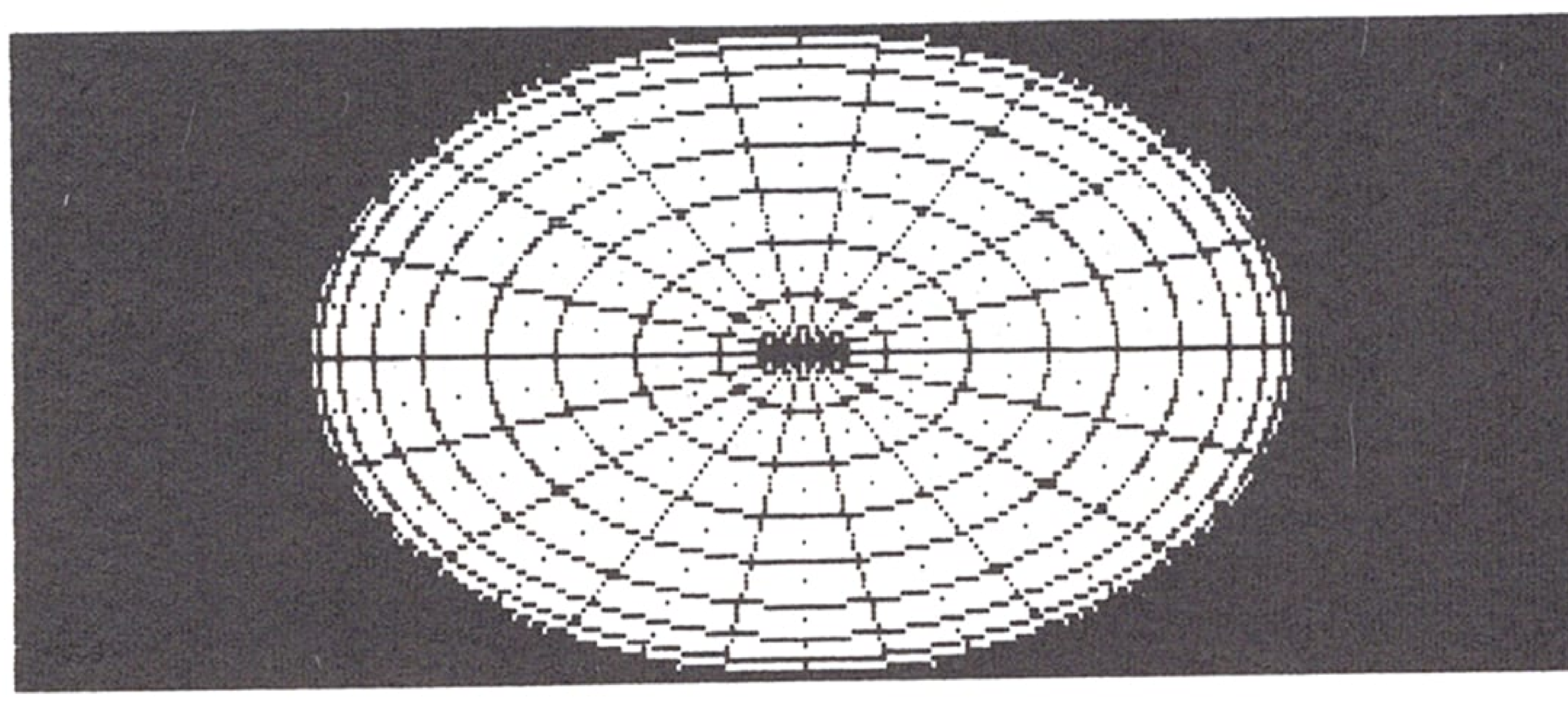
文例 PAINT (160,100),1,6



の枠で囲まれた領域』といわれても、どこだかわかりません。どこか着目点がなければ塗れないのです。これを図2のように着目点を指定してやれば、どの領域か大変はっきりします。この着目点はその領域の内部であれば、どこでも良いのです。

領域色というのは指定された領域を何色に塗るかを指定します。色はカラーパレットで指定されます。

さて手まりを塗りわけることになりますと、着目点をキチンと決めてやらねばなりません。次の図でドットを落した所が着目点となるようにします。



ここにドットが落ちるようにするにはプログラム1—13の行番号330と340の式のようにX、Y座標を選べばよいのです。あとはPAINT文の知識があれば塗りわけられます。領域色の指定方法が、

```
((I+J) MOD 3)+2
```

となっているのは赤、紫、緑が交替で出てくるようにと工夫したものです。

プログラムを走らせると数か所細かい領域でつぶせない領域が発生しますが、そこの補正までは入門書の性格上しないことにしました。

このプログラムではWINDOWやVIEW文を使っているのでプログラムが大変楽になっています。忘れた方は前にもどって読み直して下さい。

●プログラム 1—13●

```
100 '***** PROGRAM1-13*****
110 '
120 SCREEN 0,3:CLS 3:SCREEN 0,0
130 WIDTH 80,25:CONSOLE 0,25,0,0
140 WINDOW (-160,-100)-(160,100)
150 VIEW(160,50)-(480,150)
160 PI=3.14159
170 '
```

```
180 FOR I=0 TO 360 STEP 20
190 X=100*COS(I*PI/180)
200 Y=100*SIN(I*PI/180)
210 LINE (0,0)-(X,Y),6
220 NEXT I
230 '
240 FOR I=0 TO 9
250 CIRCLE (0,0),100*SIN(PI*I/18),6
260 NEXT I
270 '
280 PAINT (-160,-100),1,6
290 PAINT (160,100),1,6
300 '
310 FOR I=0 TO 360 STEP 20
320 FOR J=0 TO 9
330 X=100*SIN(PI*(J+.5)/18)*COS((I+10)*PI/180)
340 Y=100*SIN(PI*(J+.5)/18)*SIN((I+10)*PI/180)
350 PAINT(X,Y),((I+J) MOD 3)+2,6
360 NEXT J
370 NEXT I
380 '
390 END
```

さて、いよいよ手まりを回してみましょう。ここではパレットの切り換えによって、回転する感じを表現してみます。基本的なアイデアは3回を1周期として色を塗り変えることです。(プログラム1−14)

パレットのわりあては次のようになっています。

パレット番号	第1回目	第2回目	第3回目
2	紫	緑	赤
3	緑	赤	紫
4	赤	紫	緑

これが次々にくり返されれば手まりがまわり出します。

行番号410、430、450はタイマーです。あまり早くまわりすぎると、かえって回転している感じがでません。

プログラムが終った後は必ず次の短いプログラムを走らせて、パレットを初期状態に復帰させておいて下さい。

```
FOR I=1 TO 7:COLOR=(I,I):NEXT I
```

●プログラム 1—14●

```
100 '****** PROGRAM1-14******
110 '
120 SCREEN 0,0: CLS 3 :SCREEN 0,0
130 WIDTH 80,25:CONSOLE 0,25,0,0
140 WINDOW (-160,-100)-(160,100)
150 VIEW (160,50)-(480,150)
160 PI=3.14159
170 '
180 FOR I=0 TO 360 STEP 20
190 X=100*COS(I*PI/180)
200 Y=100*SIN(I*PI/180)
210 LINE (0,0)-(X,Y),6
220 NEXT I
230 '
240 FOR I=0 TO 9
250 CIRCLE (0,0),100*SIN(PI*I/18),6
260 NEXT I
270 '
280 PAINT (-160,-100),1,6
290 PAINT (160,100),1,6
300 '
310 FOR I=0 TO 360 STEP 20
320 FOR J=0 TO 9
330 X=100*SIN(PI*(J+.5)/18)*COS((I+10)*PI/180)
340 Y=100*SIN(PI*(J+.5)/18)*SIN((I+10)*PI/180)
350 PAINT (X,Y),((I+J) MOD 3)+2,6
360 NEXT J
370 NEXT I
380 '
390 FOR I=1 TO 20
400 COLOR=(2,3):COLOR=(3,4):COLOR=(4,2)
410 FOR J=1 TO 100:NEXT J
420 COLOR=(2,4):COLOR=(3,2):COLOR=(4,3)
430 FOR J=1 TO 100:NEXT J
440 COLOR=(2,2):COLOR=(3,3):COLOR=(4,4)
450 FOR J=1 TO 100:NEXT J
460 NEXT I
470 '
480 END
```

- 120〜140: 初期設定
- 150: ディスプレイ上の表示位置の指定
- 180〜220: 放射状の線を描く
- 240〜260: 同心円を描く
- 280〜290: 背景を青色に
- 350: 領域色を赤、紫、緑に
- 410: タイマー
- 390〜450: 2,3,4番のパレットをそれぞれ赤､紫､緑と交互に変化させる

第2章

簡単なアニメーションを楽しむ



2·0　SCREEN文を自由に使いこなそう

PC-８８０１は使いやすく大変良い機械です。PC-９８０１ほどではありませんが、グラフィック能力にすぐれ、インターフェース・カードも作りやすい良い機械です。

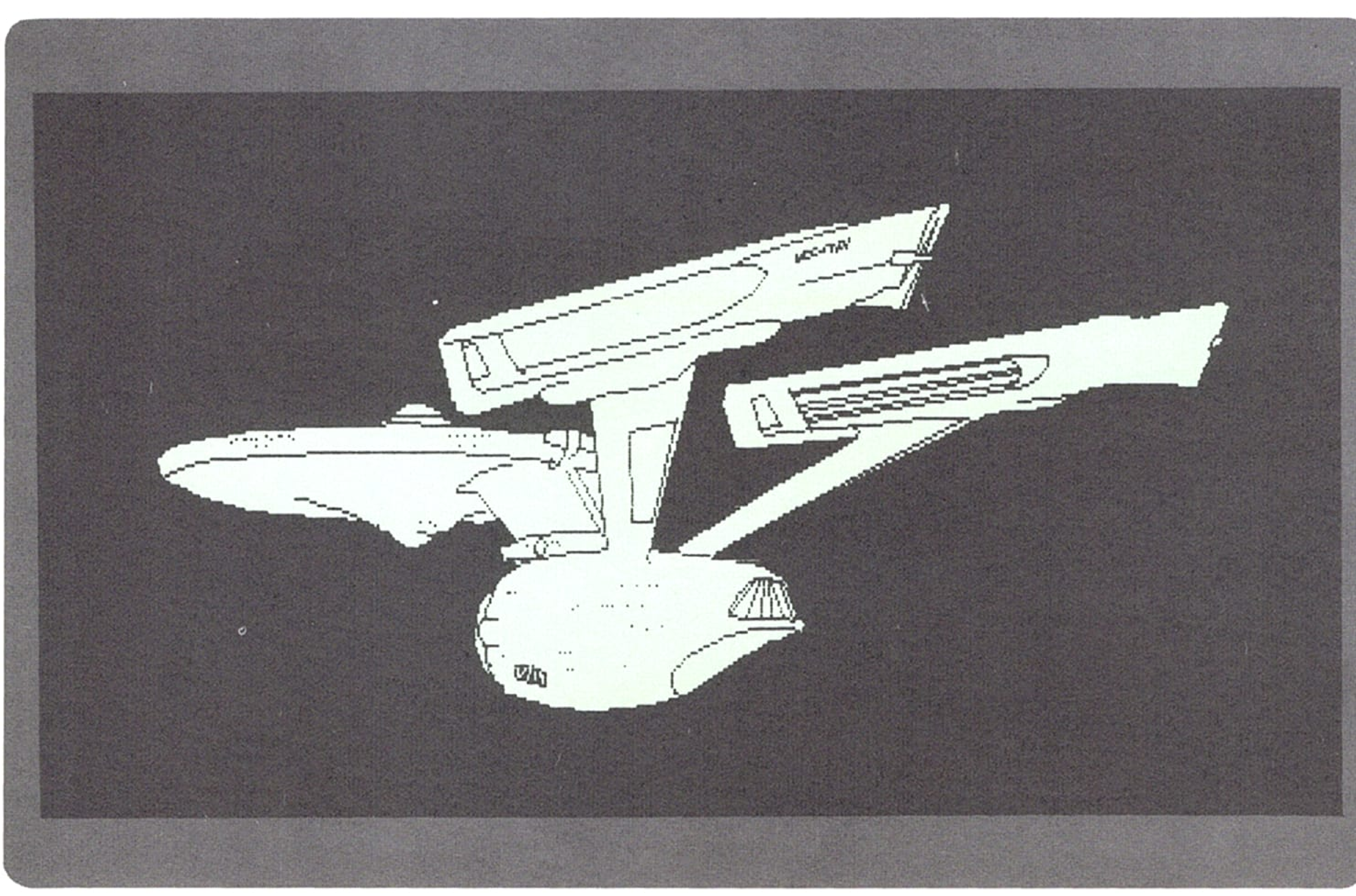
どうしたら迫真力に飛んだグラフィックスができるでしょうか。例題を通して一緒に考えてみましょう。

最初に示すのは、皆さんもよくご存知の宇宙船エンタープライズ号を入力させたものです。なかなか良くできているでしょう。入力には渡辺測器のディジタイザー、マイタブレットを使いました。

この節ではSCREEN文を上手に使って、グラフィックスを楽しむ方法について勉強することにします。

宇宙船エンタープライズ号の絵はマイコングラフィックスとしてはうまくできている方だと思いますが、皆さんの想像力はすぐに翼をひろげて、何かもっとできないかと一人歩きを始めると思います。そうです。実際その通りで何かもっと手を加えられないでしょうか？

すぐに思いつくのは星をちりばめたり、背景を入れたりすることで



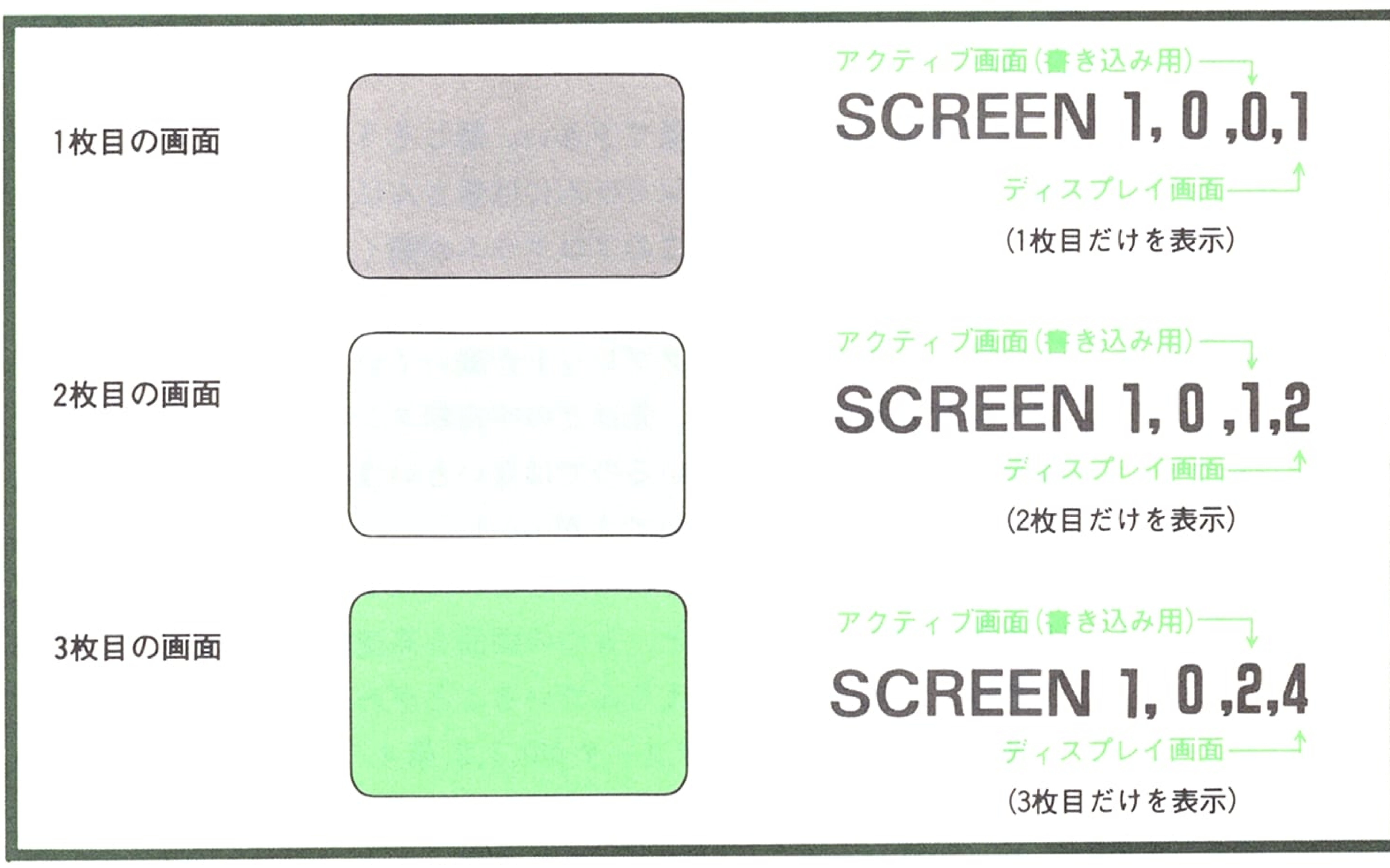
す。この画面に星を直接ちりばめるのも一つのやり方です。しかしアニメーションでよくやるように、スタートレック号と背景を分離しておいた方が、いろいろ応用がききます。そこでどうしたらそういうことができるか一緒に考えてみましょう。

SCREEN 1の場合

まず640×200モードの場合を考えます。

先にも説明しましたように、PC-８８０１の場合、単色の場合を扱うことにします。

単色の場合、画面は３枚使えました。図を入力する場合の画面の指定について復習しましょう。



それぞれの画面に入力するには上のようなSCREEN文を使うと便利です。アクティブ画面という書き込み用の画面は３枚しかありませんので、許されるパラメータは0､1､2の３つしかあまりませんが､ディスプレイ画面のパラメータは0から7までの８通りあります。このうちディスプレイ画面のパラメータは、

第１画面だけを表示したいとき——1
第２画面だけを表示したいとき——2
第３画面だけを表示したいとき——4

とえらびます。残りの0､3､5､6､7は何だったでしょう。ついでに復習しておきましょう。

何も表示しないとき――――――――――0
第1画面と第2画面を合成して表示したいとき――3
第1画面と第3画面を合成して表示したいとき――5
第2画面と第3画面を合成して表示したいとき――6
第1､2､3 画面を全て合成して表示したいとき――7

とえらぶのでした。

別々の画面に書き込むときは他の画面が見えない方が良いのです。なぜなら、別の画面が見えていると、これは一体どちらの画面に書いたかわからないというものが出てきてしまうからです。そういうわけで、書き込み時には、第4パラメータには 1､2､4 を選ぶのが良いと思います。それでは皆さんと一緒に画面に絵を描いて行くことにしましょう。

まず次のプログラムを見て下さい。難しそうだなと思われるでしょう。そして実際、このプログラムには皆さんに全く説明していないテクニックがあり、そしてこのプログラムが動くためには、あらかじめ別のプログラムがあって、作業をしています。しかもその際フロッピー・ディスク 装置やマイタブレットが働いているのです。あえてプログラムをお見せするのは、先ほどの宇宙船エンタープライズ号は簡単なプログラムで書かれているのではないということを知っていただくためです（長さは大変短いですが……）。

ここで皆さんに注目していただきたいのはプログラム中の色分けをした部分です。よく見ると、3つの画面を高速消去してから、1枚目、2枚目、3枚目の画面をえらんでいることがわかるでしょう。

SCREEN文の第3パラメータが0､1､2､第4パラメータが1､2､4であることに注意して下さい。

ここを理解してもらいたかったのです。

〈注〉下のプログラムは説明のためのサンプルプログラムです。入力せずにプログラムを追ってみて処理過程を理解してください。

●サンプル・プログラム●

第1画面への入力

```
100 FOR I=1 TO 7:COLOR=(I,I):NEXT
110 SCREEN 1,3,0,1:CLS 3:SCREEN 1,0,0,1
120 CONSOLE 0,25,0,1:WIDTH 80,25
130 COLOR 7,0,0,7
140 COL=7
150 WINDOW (0,0)-(639,399):VIEW(0,0)-(639,199)
160 '
170 OPEN "STATREK" FOR INPUT AS #1
180 IF EOF(1) THEN GOTO *HORIZONT
```

●第1画面の部分はあらかじめデータファイルSTARTREKがディスケットに設定していないと動きません。

```
190 INPUT #1,X,Y,F
200 X=INT(X/6.72+.5):Y=399-INT(Y/7+.5)
210 IF PCNT=0 OR F<>1 THEN PSET(X,Y),COL:GOTO
    230
220 LINE(XOLD,YOLD)-(X,Y),COL
230 XOLD=X:YOLD=Y
240 PCNT=1
250 GOTO 180
260 '
270 *HORIZONT
280 SCREEN 1,3,1,2:CLS 3:SCREEN 1,0,1,2
290 WINDOW(0,0)-(639,399):VIEW(0,0)-(639,199)
300 Y1=100:Y2=300:D1=1:C=0
310 FOR I=1 TO 50
320 LINE(0,Y1)-(639,Y1)  ┐
330 LINE(0,Y2)-(639,Y2)  ┘― 等比級数的に横の線を引く
340 Y1=Y1-D1:Y2=Y2+D1:D1=D1*1.1
350 NEXT I
360 '
370 FOR I=-56 TO 56
380 U(0)=160*I
390 U(1)=160*(-I)
400 LINE(U(0),0)-(U(1),399)
410 NEXT I
420 '
430 LINE(0,101)-(639,299),0,BF
440 '
450 SCREEN 1,3,2,4:CLS 3:SCREEN 1,0,2,4
455 WINDOW(0,0)-(639,399):VIEW(0,0)-(639,199)
460 FOR I=1 TO 50
470 X=RND*639:Y=RND*200+100
480 R=1+.01*RND*SQR(X^2+Y^2)
490 C=INT(RND*7+1)
500 CIRCLE(X,Y),R,C ←― いん石を描く
505 D=1
510 CIRCLE(X-D,Y),R+D,C ←― いん石にカゲをつける
520 PAINT(X,Y),C
530 NEXT I
540 '
550 SCREEN 0,0 ←――― 青、赤、緑のプレーンを合成してカラー表示する
```

第2画面への入力（290〜440）

ワールド座標上の点(-8960,0)と点(8960,399)を起点として、それぞれ反対方向に移動させながら2点を結ぶ直線を引く（370〜410）

第3画面への入力（455〜530）

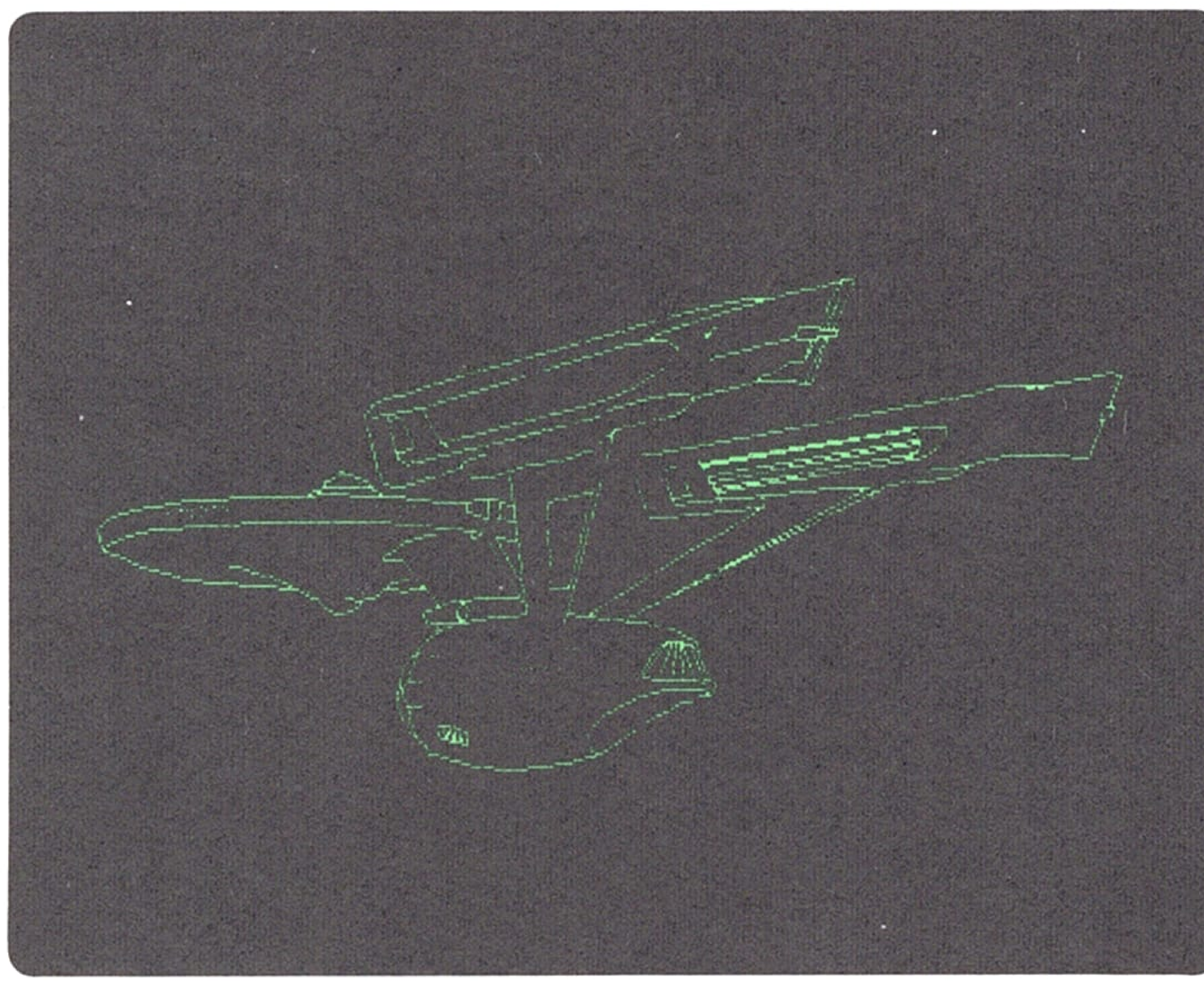
第1画面への入力

プログラムの行番号180から250までの部分で、PC-8801はマイタブレットから入力され、フロッピー・ディスクにしまわれたデータを読み出してきては画面に表示しています。その結果エンタープライズ号が画面に現われてくるわけです。

ディジタイザーを使わないで、図形を入力する方法については後で勉強することにしましょう。今はとりあえずSCREEN文の勉強です。

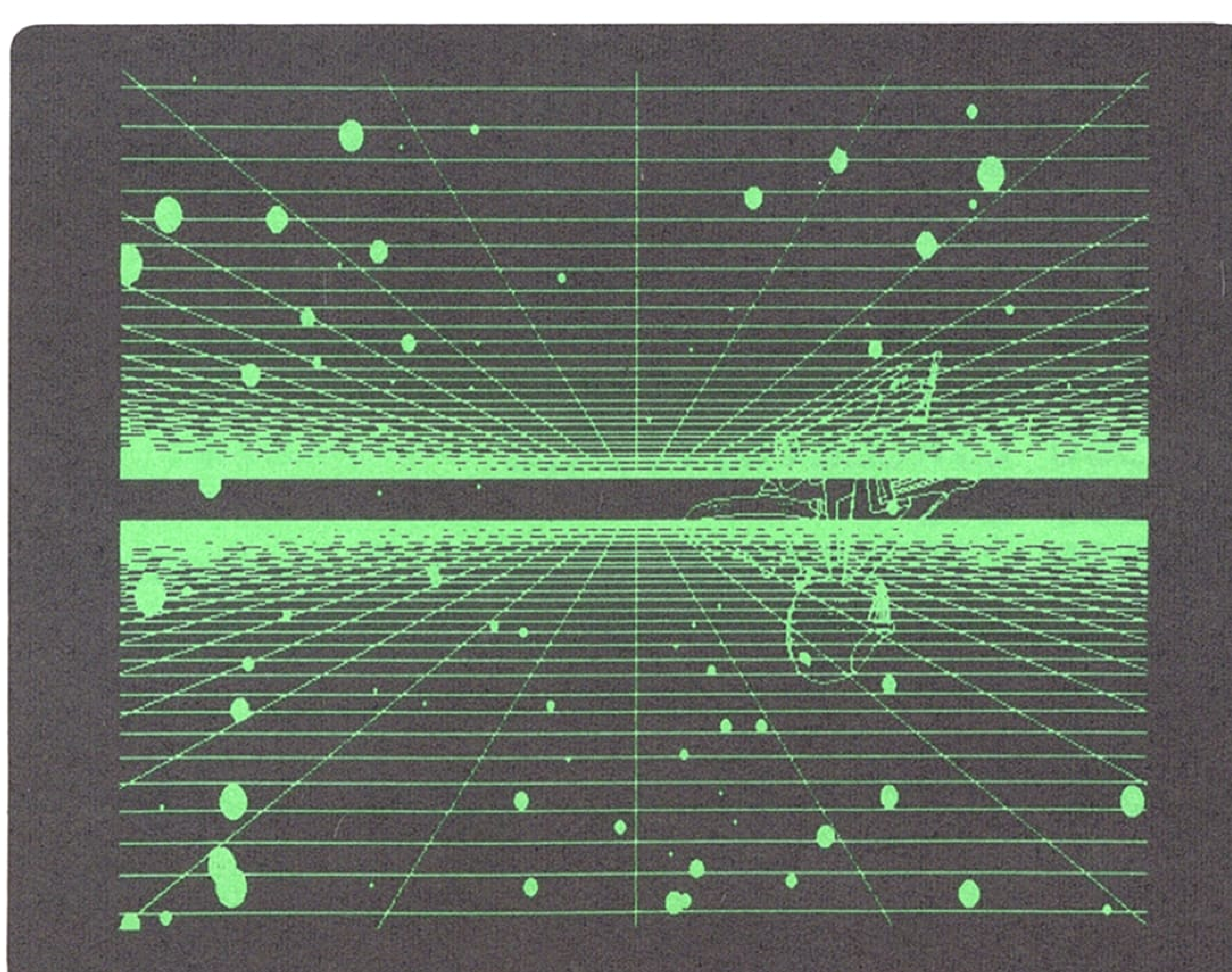
SCREEN 1, 0 ,0,1

第１画面にあらわれる宇宙船エンタープライズ号→



STAR GATE 通過中の宇宙船エンタープライズ号→

●これが本来のシーンですが、プリンタにおとす都合上プログラムを変更しました。

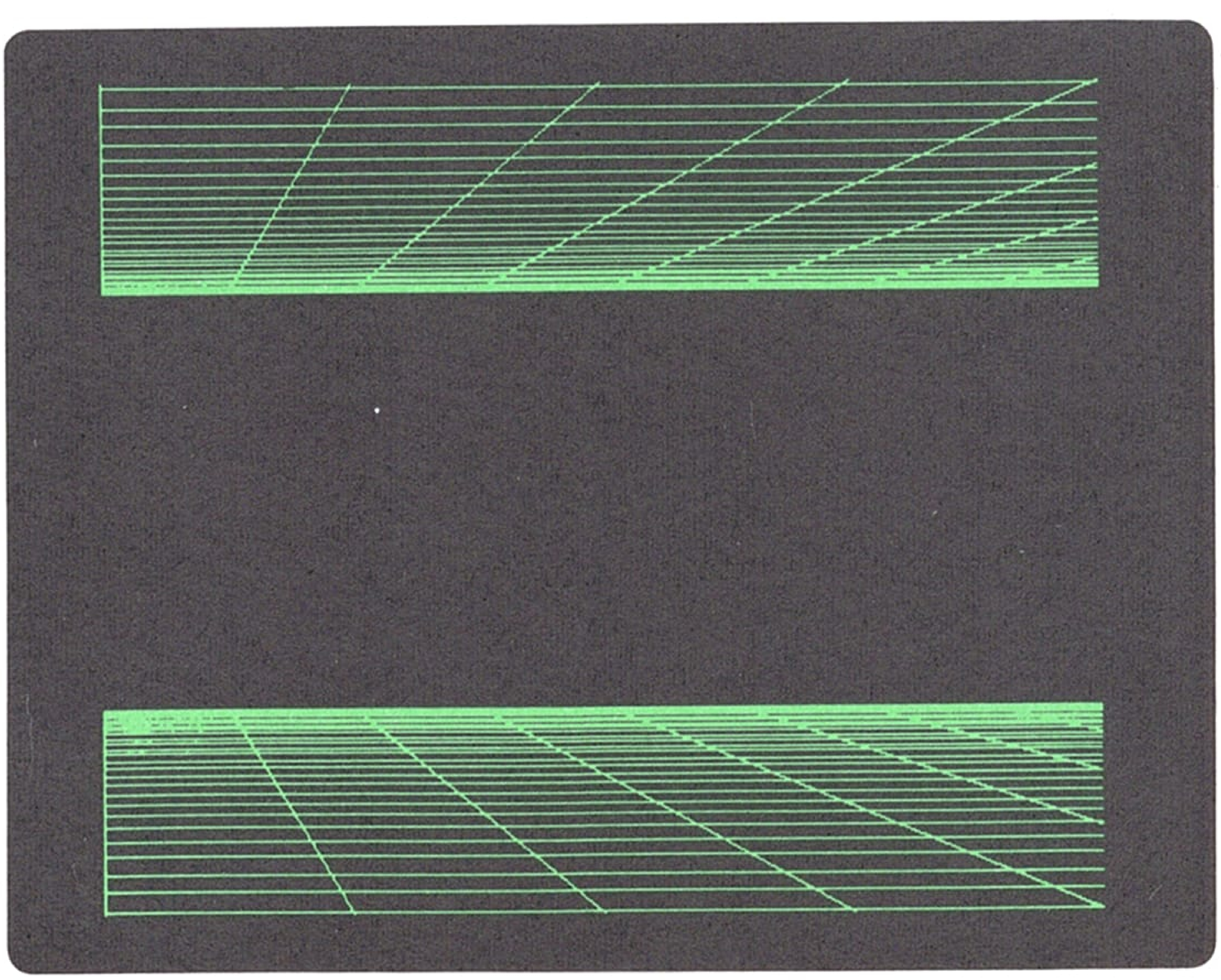


第2画面への入力

第１画面には宇宙船エンタープライズを入力しました。第２画面には背景を入れようと思います。最初の予定では、『2001年宇宙の旅』のラストシーンでででてきた STAR GATE という光の壁のようなものをつけてしまおうとやってみました。かなり、うまくできたと思うのですが（前ページ下図）、プリンターにおとすと色わけできないために宇宙船エンタープライズ号がハッキリ見えないので、光の壁を少し離して図柄がかさならないようにしました。何とも残念です。第２画面に図を描くためのプログラムをかかげます。

〈注〉下のプログラムは参考のためにかかげました。入力する必要はありません。

```
280 SCREEN 1,3,1,2:CLS 3:SCREEN 1,0,1,2
290 WINDOW(0,0)-(639,399):VIEW(0,0)-(639,199)
300 Y1=100:Y2=300:D1=1:C=0
310 FOR I=1 TO 50
320 LINE(0,Y1)-(639,Y1)
330 LINE(0,Y2)-(639,Y2)
340 Y1=Y1-D1:Y2=Y2+D1:D1=D1*1.1
350 NEXT I
360 '
370 FOR I=-56 TO 56
380 U(0)=160*I
390 U(1)=160*(-I)
400 LINE(U(0),0)-(U(1),399)
410 NEXT I
420 '
430 LINE(0,101)-(639,299),0,BF
```

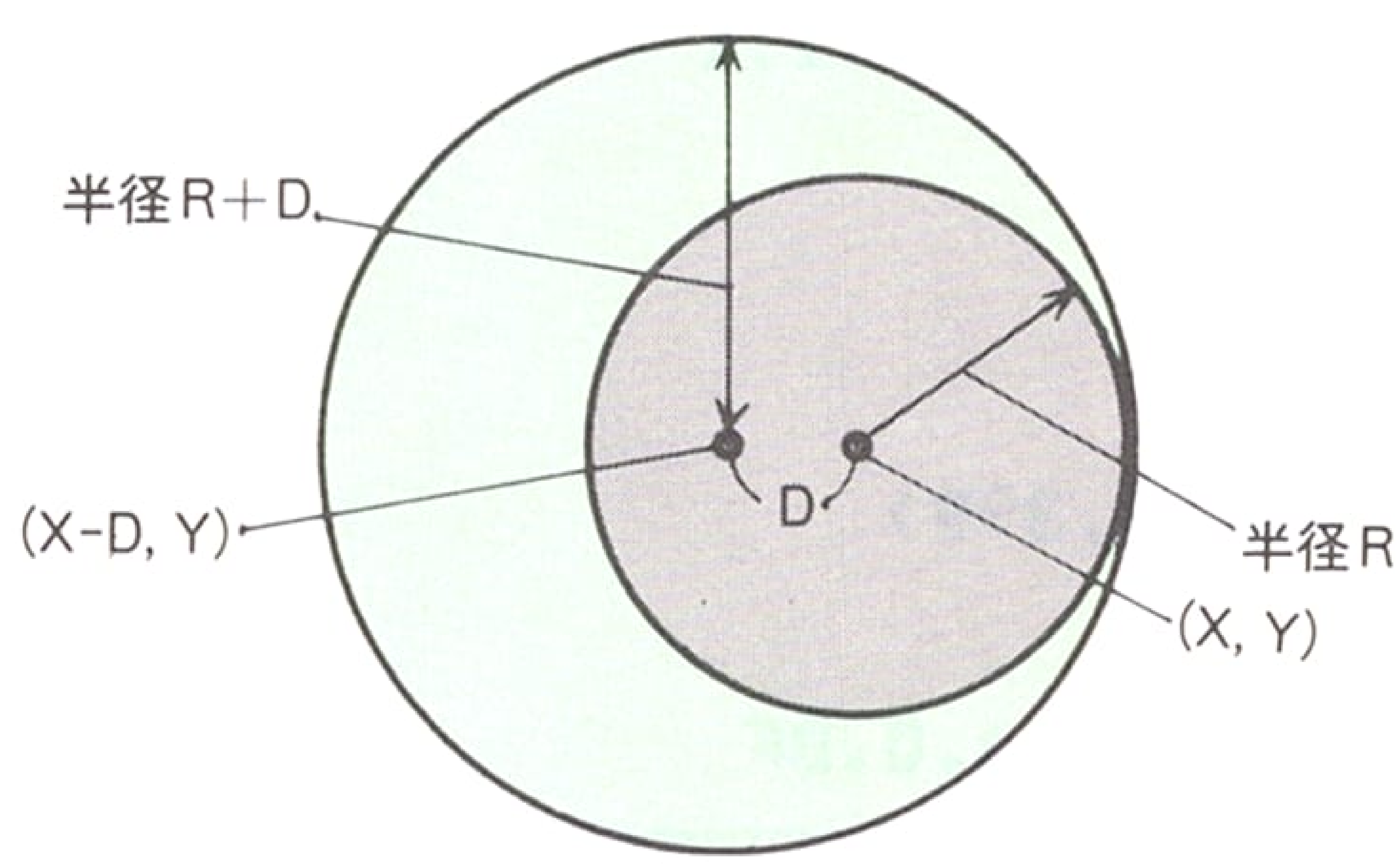


SCREEN 1,0,1,2

この図は何か見覚えがあるでしょう。最近よくテレビのコマーシャルで、地平線の彼方から大地が動いてくるようなやつです。これはパレット文を使ってきわめて簡単に動かすテクニックがあります。

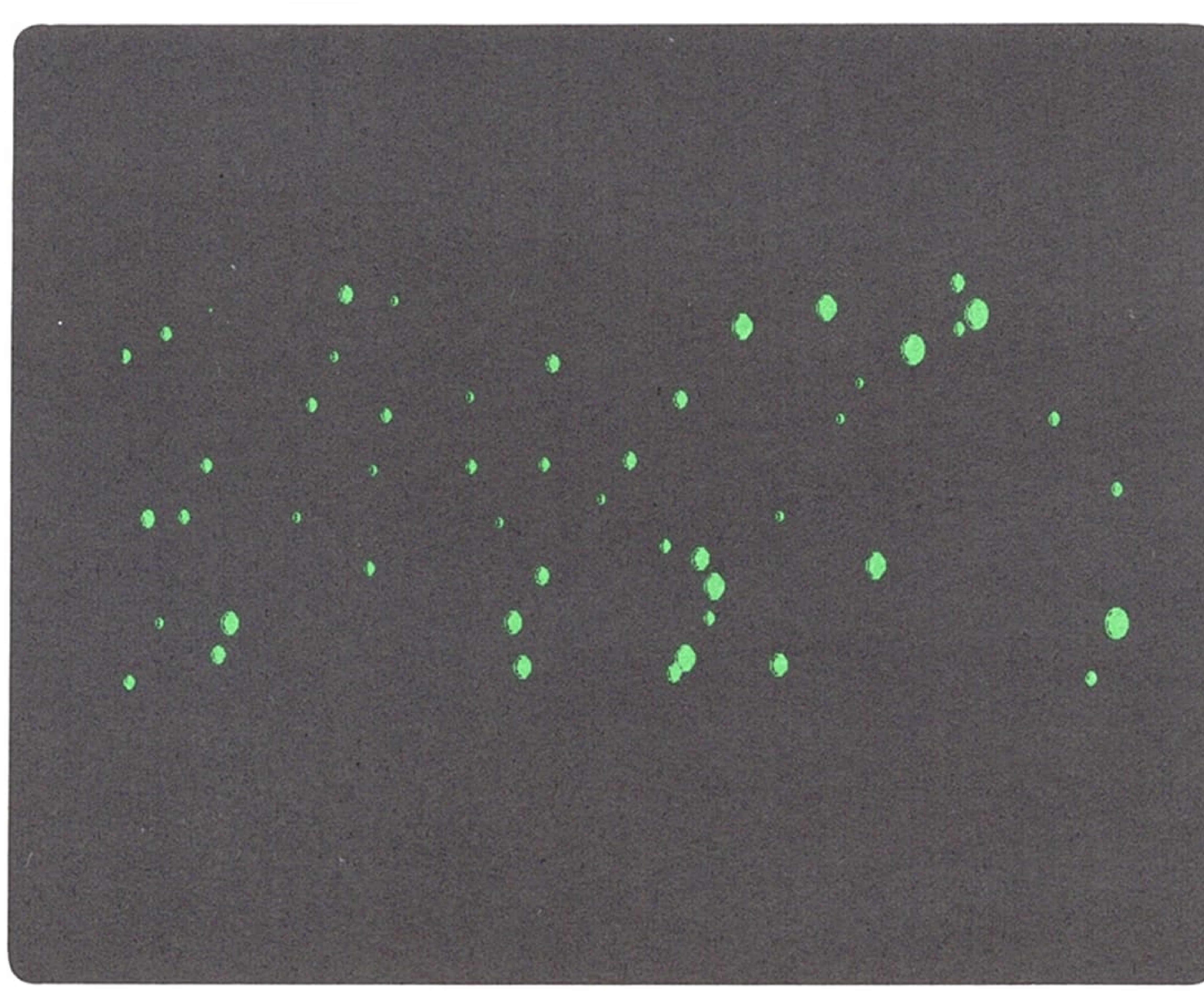
この部分のプログラムには何も難しいところはありません。テクニックとしては遠近感を出すための横線の間隔が1.1ずつの等比列になっているだけです。コマーシャルでは超大型コンピュータで作りましたという感じの説明をしていますが、PC-8801の方がずっと行き届いたグラフィックスができるのです。

さて、第3画面は空間に浮いている、いん石を作ることにします。いん石の中心位置を乱数で適当にばらまき、視野の中心から遠ざかるに従って、いん石の大きさを大きくするようにすれば、それらしくなるでしょう。この場合、いん石にリアリティを持たせようと、影をつける工夫をしてみました。イラストでよくやる手です。59ページのプログラムの行番号510がそれです。



●行番号500で半径Rの円を描き、行番号510で中心をDだけ移動して半径R+Dの円を描いている

SCREEN 1,0,2,4



さて、これで、次のようにわりふりができました。

第1画面	宇宙船エンタープライズ号	SCREEN 1,0,0,1
第2画面	STAR GATE	SCREEN 1,0,1,2
第3画面	いん石群	SCREEN 1,0,2,4

もう、これ以上はかきこめません。しかし合成表示が残っていました。これをやってみるとどうなるか、ためしてみましょう。

❶第1画面と第2画面の合成表示

すでに画面に書き込みが終了した今、アクティブ画面のパラメータによる指定は不用です。そこで第3パラメータは省略します。第2パラメータも省略します。必要なのは第4パラメータの3だけです。

SCREEN 1, , ,3

こうしますと、第1画面の宇宙船エンタープライズ号と第2画面のSTAR GATEが合成されて表示されます。とても面白いですね。

❷第1画面と第3画面の合成表示

次に第1画面の宇宙船エンタープライズ号と第3画面のいん石群とを合成するにはどうしたら良いでしょうか？　そうです。SCREEN文の第4パラメータを5にすればよかったのです。

SCREEN 1, , ,5

エンタープライズ号がいん石群の中をすすんでる感じが、いくらかでるでしょう。

❸第2画面と第3画面の合成表示

それでは第2画面のSTAR GATEと第3のいん石群を合成するにはどうしたら良いでしょう。もうわかりますね。第4パラメータを6にするだけです。

SCREEN 1, , ,6

スタンリー・キューブリックの世界にはほど遠いのですが、それでも何とか気分だけは楽しめるのではないでしょうか。

❹全画面の合成表示

最後に第1画面、第2画面、第3画面の全てを合成するにはどうしたらよいでしょう。そうです。次の通りでした。

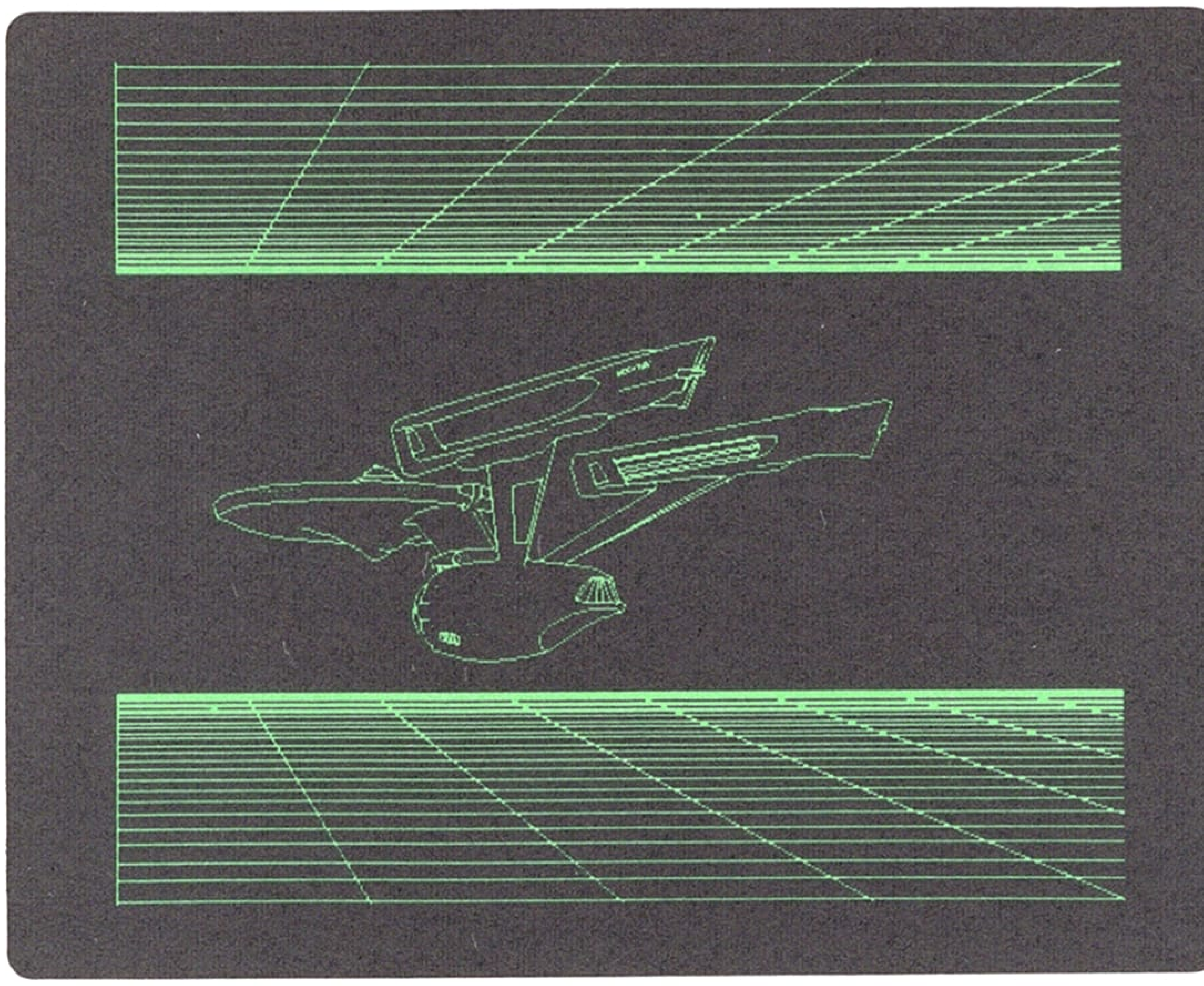
SCREEN 1, , ,7

これで所期の画面はでき上りました。合成して気に入らない時は、第1画面だけ、第2画面だけ、第3画面だけを各々独立に修正できます。これが1枚画面ではない長所なのです。1枚の画面ですと修正の

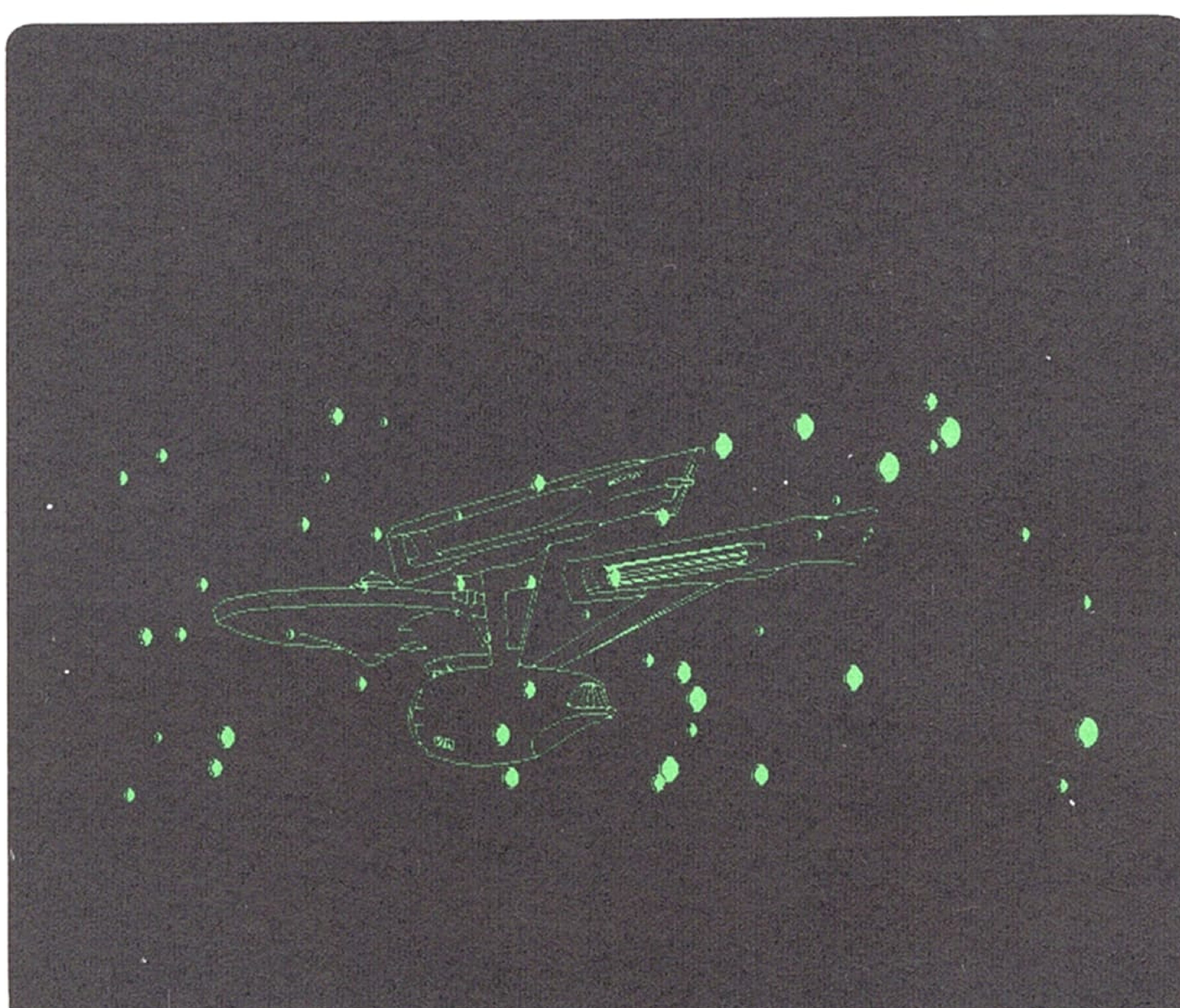
第4パラメータ	ページ3	ページ2	ページ1
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

●第4パラメータを3ビットの2進表示にした場合、1が表示、0が無表示のページになります。

SCREEN 1, , ,3



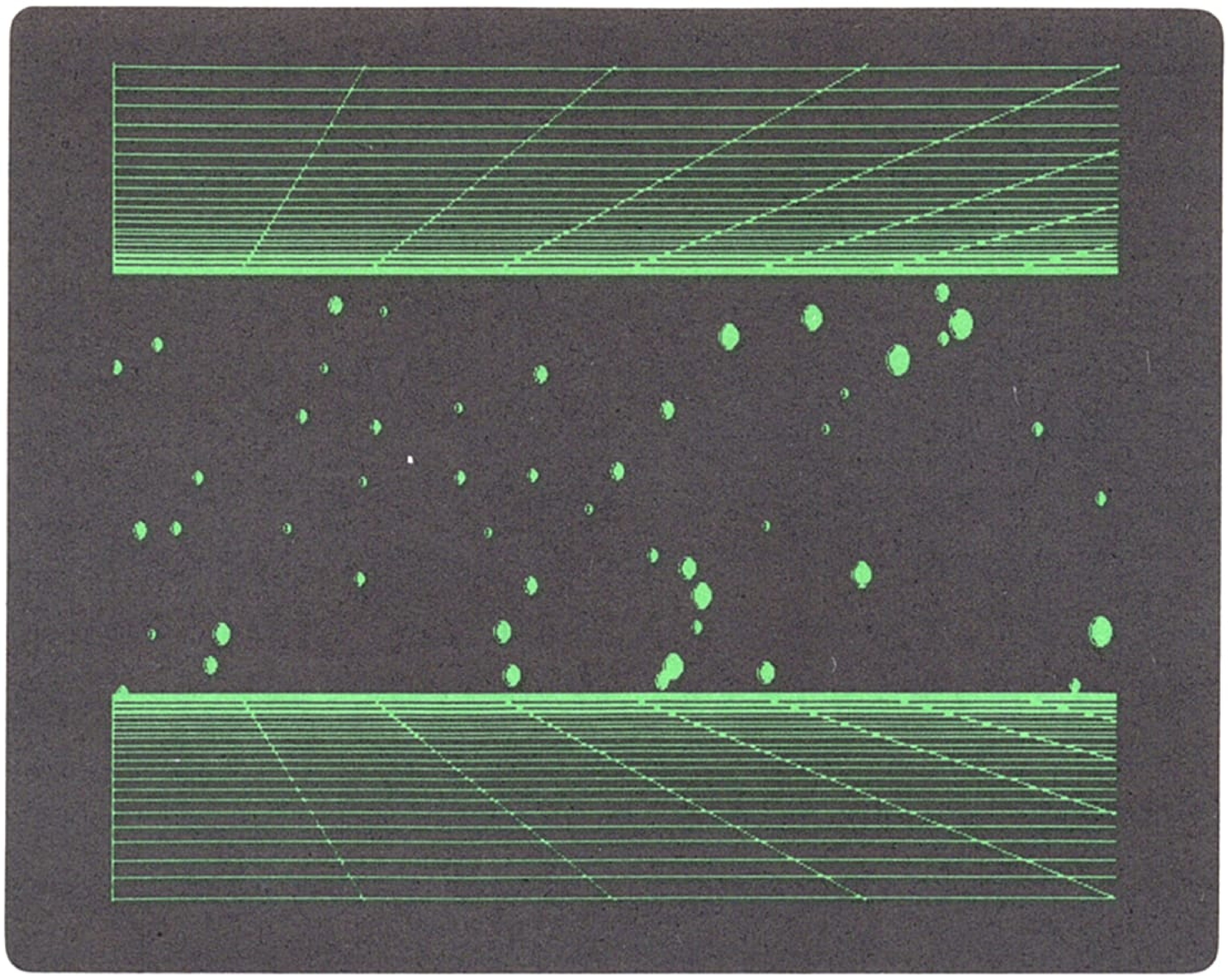
SCREEN 1, , ,5



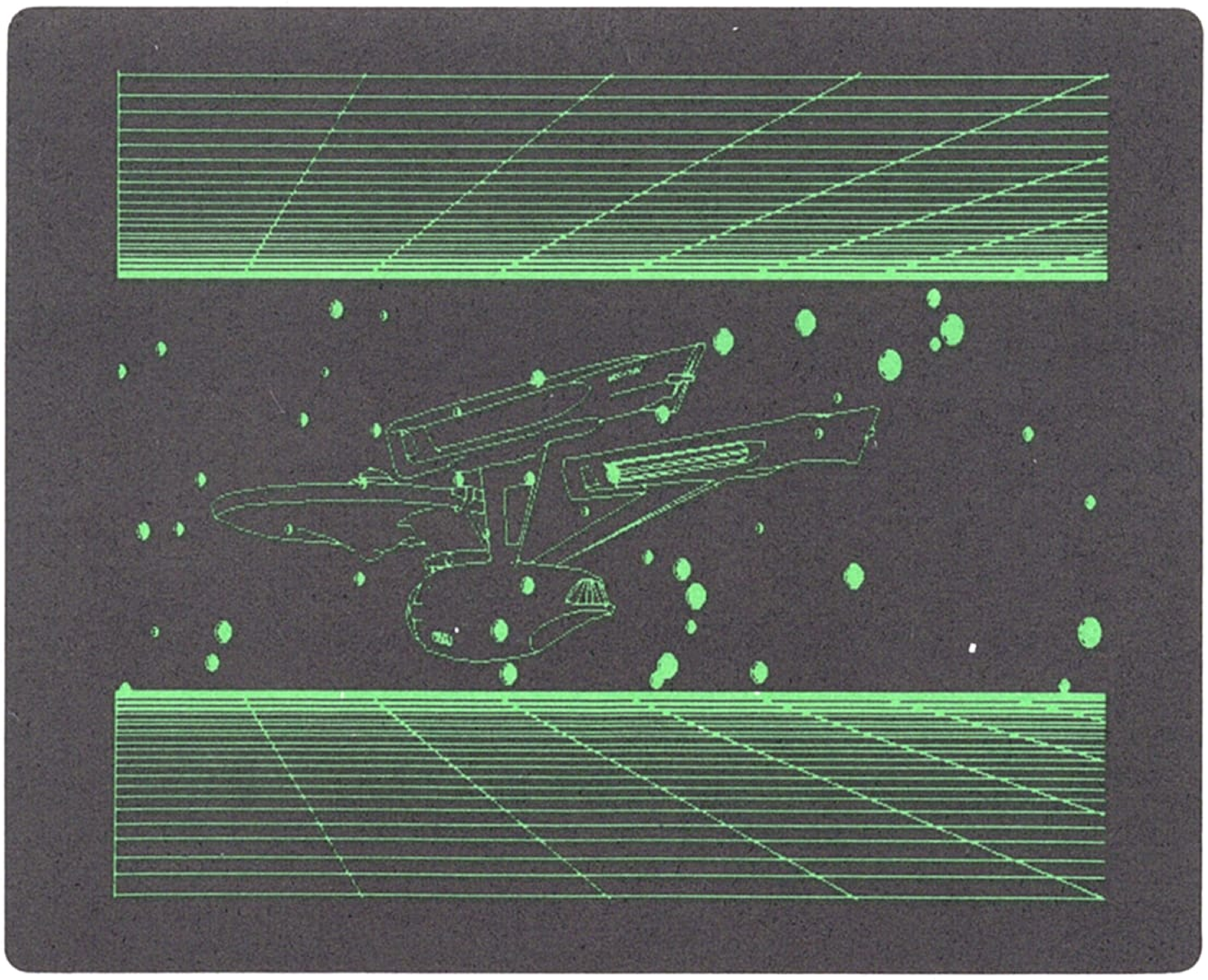
際にエンタープライズ号がこわれてしまったり、いろいろ悲劇の起る可能性があります。

４つの合成画面表示を64～65ページに示します。

しかし、これでまだすんだわけではありません。あなたが全ての絵を入力したつもりになって下さい。全て単色のグリーンで表示されて



SCREEN 1, , ,6



SCREEN 1, , ,7

いますね。そこで、次のように入力したと考えて下さい。

SCREEN　0,0

どうなるでしょう。あざやかにカラーで像が浮び上がってきますね。第１画面は青の表示、第２画面は赤の表示、第３画面は緑の表示で全てが合成されて表示されます。何となくカラーグラフィックスの秘密がわかってきたでしょう。

2·1 エンタープライズ号をDATA文で描く

ナツメ社からの御希望でエンタープライズ号をフロッピーディスクを使わずに描いてみる方法について考えてみました。途方もない精力が必要だと思うのですが、案外入力してしまう人もいるのではないかという意見もありましたので、フロッピーディスクからデータを読み出してみました。これを印字した結果を示すことにしました。

これらの読み出しデータをどう利用するかについて簡単に説明しておきましょう。フロッピーディスクを使わないのですから、READ～DATA文を使う他はありません。この方法はINPUT＃の代りにREAD文を置き、プログラムのうしろにDATA文をつけることになります。DATAの後に並べて行くのです。誤りがあった時にチェックしやすいように、3個ずつ1まとめにして、1行中には9個とか15個とか並べるようにしておいて下さい。ここでのプログラムでは先頭のデータだけを並べてあります。全てのデータを入力したら、最後のデータは0，0,0としておいて下さい。こうすれば最後のデータの検出になります。根気よく入力していけばできるかも知れません。プログラムは簡単なので、楽だとは思います。

☞ 3 (P.212)

●宇宙船エンタープライズ・プログラムリスト●

```
100 FOR I=1 TO 7:COLOR=(I,I):NEXT I  ← カラーパレットの初期化
110 SCREEN 0,3:CLS 3:SCREEN 0,0  ← 640×200のカラーモード
120 CONSOLE 0,25,0,1:WIDTH 80,25
130 COLOR 7,0,0,7
140 COL=7
150 WINDOW(0,0)-(639,199):VIEW(0,0)-(639,199)
160 READ X,Y,F  ← データを3つずつ読み込む
170 IF F=0 THEN GOTO *FINISH  ← F=0なら終り
180 X=INT(X/6.72+.5):Y=299-INT(Y/7+.5)          ┐
190 IF PCNT=0 OR F<>1 THEN PSET(X,Y),COL:GOTO 210 ├ 画面を描く
200 LINE(XOLD,YOLD)-(X,Y),COL                   ┘
210 XOLD=X:YOLD=Y
220 PCNT=1
230 GOTO 160
240 *FINISH
250 END
260 '
270 DATA 425,1290,2,413,1330,1,427,1343,1  ┐
280 DATA 459,1357,1,486,1363,1,520,1374,1  ├ 67～68ページのデータをこのように入れてください。
290 DATA 569,1380,1,623,1388,1,665,1393,1  ┘
300 DATA 000,0000,0
```

●宇宙船エンタープライズ・データ●

425	1290	2
413	1330	1
427	1343	1
459	1357	1
483	1363	1
520	1374	1
569	1380	1
632	1388	1
665	1393	1
715	1394	1
767	1395	1
808	1397	1
851	1398	1
905	1399	1
950	1401	1
976	1401	1
1014	1402	1
1058	1405	1
1106	1404	1
1144	1403	1
1182	1403	1
1234	1399	1
1271	1397	1
1308	1394	1
1344	1391	1
1378	1386	1
1412	1385	1
1441	1383	1
1451	1383	1
1467	1389	1
1487	1391	1
1503	1394	1
1519	1395	1
1534	1393	1
1547	1391	1
1558	1387	1
1569	1384	1
1577	1382	1
1589	1377	1
964	1197	4
944	1199	1
904	1202	1
863	1202	1
829	1202	1
789	1208	1
754	1210	1
729	1213	1
692	1217	1
663	1219	1
635	1225	1
602	1234	1
569	1240	1
537	1245	1
545	1243	1
511	1255	1
482	1265	1
459	1275	1
440	1279	1
429	1282	1
427	1287	1
430	1291	1
438	1297	1
455	1304	1
477	1310	1
498	1316	1
533	1323	1
576	1328	1
602	1330	1
635	1332	1
680	1335	1
714	1336	1
756	1336	1
786	1337	1
856	1343	1
895	1345	1
929	1345	1
962	1344	1
993	1345	1
1015	1345	1
1035	1345	1
1058	1345	1
1085	1345	1
1112	1345	1
1148	1344	1

1181	1342	1
1209	1343	1
1238	1341	1
1264	1338	1
1290	1338	1
1313	1336	1
1335	1335	1
1359	1334	1
1386	1331	1
1412	1329	1
1439	1328	1
1458	1323	1
1472	1323	1
1482	1323	1
789	1244	4
812	1243	1
831	1241	1
862	1234	1
887	1228	1
903	1222	1
925	1216	1
944	1210	1
958	1203	1
973	1198	1
982	1195	1
997	1187	1
1009	1180	1
1023	1174	1
1033	1174	1
1039	1168	1
1048	1162	1
1061	1154	1
1073	1148	1
1082	1143	1
1100	1143	1
1117	1143	1
1130	1146	1
1139	1151	1
1146	1158	1
1153	1162	1
1156	1170	1
1116	1173	4
1129	1173	1
1147	1172	1
1160	1172	1
1172	1173	1
1184	1173	1
1193	1176	1
1209	1182	1
1218	1188	1
1228	1196	1
1239	1203	1
1251	1208	1
1266	1213	1
1278	1219	1
1297	1221	1
1226	1194	4
1253	1195	1
1284	1192	1
1307	1196	1
1330	1197	1
1342	1197	1
1373	1318	4
1342	1315	1
1316	1308	1
1293	1302	1
1267	1292	1
1244	1277	1
1232	1271	1
1220	1260	1
1232	1260	4
1257	1260	1
1278	1257	1
1295	1241	1
1320	1216	1
1341	1198	1
1364	1179	1
1370	1168	1
1376	1159	1
1374	1119	4
1376	1101	1
999	1402	4
1024	1413	1
1037	1417	1

1053	1422	1
1073	1423	1
1096	1424	1
1115	1424	1
1135	1422	1
1166	1421	1
1184	1421	1
1196	1415	1
1205	1412	1
1211	1409	1
1218	1404	1
1222	1400	1
1046	1423	4
1057	1432	1
1069	1438	1
1082	1442	1
1092	1448	1
1103	1448	1
1118	1449	1
1130	1448	1
1146	1444	1
1159	1443	1
1172	1438	1
1177	1434	1
1184	1426	1
1187	1418	1
1093	1452	4
1096	1455	1
1100	1458	1
1104	1458	1
1108	1456	1
1114	1451	1
1460	1385	4
1457	1366	1
1521	1349	1
1519	1395	4
1519	1348	1
1507	1332	1
1497	1318	1
1473	1306	1
1480	1328	1
1555	1389	4
1549	1348	1
1523	1330	1
1548	1348	2
1597	1347	1
1601	1303	2
1542	1310	1
1619	1200	1
1520	1301	2
1615	1172	1
1621	1171	2
1593	1173	1
1554	1173	1
1519	1173	1
1484	1173	1
1457	1171	1
1425	1167	1
1391	1159	1
1375	1158	1
1358	1151	1
1347	1147	1
1337	1142	1
1343	1116	1
1368	1118	1
1384	1118	1
1418	1119	1
1579	1456	2
1625	1138	1
1584	1137	1
1530	1135	1
1490	1130	1
1437	1122	1
1403	1113	1
1374	1099	1
1347	1083	1
1323	1063	1
1317	1048	1
1310	1032	1
1299	1014	1
1295	1004	1
1293	992	1
1332	995	2
1317	997	1

1308	996	1
1299	994	1
1291	991	1
1282	984	1
1279	977	1
1278	969	1
1278	959	1
1281	951	1
1287	946	1
1294	939	1
1302	937	1
1312	937	1
1322	938	1
1328	938	1
1292	939	2
1294	925	1
1297	906	1
1303	892	1
1311	879	1
1319	868	1
1330	860	1
1341	853	1
1346	840	1
1355	835	1
1371	833	1
1392	824	1
1422	817	1
1455	811	1
1483	804	1
1518	802	1
1555	799	1
1584	799	1
1613	799	1
1648	802	1
1684	806	1
1718	809	1
1739	815	1
1772	819	1
1811	825	1
1841	830	1
1858	839	1
1880	853	1
1906	866	1
1932	882	1
1962	896	1
1984	909	1
2003	918	1
2020	924	1
2042	932	1
2063	942	1
2089	948	1
2108	954	1
2121	958	1
2133	960	1
2141	961	1
2146	965	1
2146	967	1
2141	968	1
2136	968	1
2127	970	1
2111	971	1
2090	973	1
2071	971	1
2021	972	1
1989	966	1
1947	957	1
1912	948	1
1891	939	1
1863	923	1
1842	914	1
1825	902	1
1811	891	1
1803	885	1
1801	873	1
1798	866	1
1799	859	1
1802	851	1
1808	843	1
1816	838	1
1824	835	1
1842	835	1
1325	1059	4
1307	1052	1
1288	1033	1

1280	1020	1
1267	990	1
1264	964	1
1263	945	1
1268	924	1
1278	912	1
1290	891	1
1298	879	1
1314	871	1
2151	959	4
2165	984	1
2145	991	1
2120	1043	1
2120	1054	1
2117	1064	1
2107	1076	1
2093	1086	1
2069	1094	1
2044	1104	1
2003	1111	1
1865	1128	1
1776	1136	1
1751	1140	1
1742	1142	1
1740	1113	4
1736	1118	1
1735	1120	1
1735	1127	1
1740	1134	1
1877	1544	1
1793	1136	2
2099	1282	1
2258	1354	1
2282	1368	1
2295	1375	1
2301	1379	1
2303	1384	1
2303	1388	1
2301	1391	1
2293	1396	1
1914	1119	2
1918	1126	1
1923	1132	1
1933	1136	1
1945	1143	1
2597	1413	1
1940	1143	2
2656	1420	1
2715	1425	1
2770	1429	1
2811	1434	1
2830	1437	1
2845	1440	1
2866	1447	1
2878	1458	1
2886	1466	1
2122	1037	2
2114	1053	1
2107	1060	1
2096	1068	1
2091	1072	1
2076	1074	1
2060	1076	1
2040	1075	1
2024	1075	1
2009	1068	1
1997	1058	1
1981	1049	1
1967	1032	1
1959	1020	1
1954	1012	1
1954	1001	1
1965	1001	1
1976	999	1
1991	997	1
2005	997	1
2036	994	1
2072	995	1
2109	992	1
2128	990	1
2145	991	1
2025	1072	2
1958	1001	1
2036	1073	4

2002	995	1	2531	1875	4	2018	1617	1	2814	1544	1	1245	1381	4
2051	1076	4	2432	1858	1	2045	1622	1	2816	1539	1	1267	1379	4
2042	994	1	2371	1845	1	2065	1622	1	2817	1534	1	1290	1377	4
2062	1076	4	2318	1831	1	2087	1622	1	2815	1525	1	1309	1377	4
2073	995	1	2257	1820	1	2099	1618	1	2808	1516	1	1233	1360	4
2071	1074	4	2190	1805	1	2110	1615	1	2796	1506	1	1273	1358	4
2102	995	1	2134	1792	1	1238	1569	4	2780	1498	1	1668	1064	4
2087	1073	4	2077	1778	1	1271	1574	1	2766	1493	1	1705	1061	4
2128	992	1	2032	1766	1	1311	1515	1	2753	1487	1	1741	1063	4
1516	1494	2	1997	1754	1	1308	1509	1	2736	1480	1	1763	1063	4
1495	1489	1	1966	1741	1	1304	1507	1	2728	1478	1	1534	1017	4
1478	1486	1	2316	1831	4	1261	1496	1	2711	1474	1	1548	1021	4
1462	1481	1	2300	1835	1	1812	1412	4	2683	1470	1	1612	1020	4
1453	1474	1	2282	1835	1	1745	1200	1	2165	1404	1	1626	1022	4
1444	1468	1	2251	1834	1	1694	1196	1	3271	1587	4	1642	1022	4
1442	1464	1	2210	1829	1	1685	1388	1	3255	1560	1	1679	1024	4
1446	1460	1	2165	1820	1	1811	1410	1	3284	1566	1	1717	1025	4
1456	1452	1	2124	1815	1	2606	1396	4	3294	1588	1	1694	978	4
1477	1443	1	2056	1801	1	2606	1409	1	3273	1589	1	1732	976	4
1500	1442	1	1986	1782	1	2600	1412	1	2699	1544	4	1772	977	4
1530	1445	1	1345	1643	1	2484	1414	1	2714	1537	1	1496	924	4
1558	1450	1	1301	1633	1	2332	1400	1	2725	1535	1	1510	924	4
1637	1467	1	1261	1623	1	2100	1383	1	2735	1526	1	1523	925	4
1729	1479	1	1227	1614	1	2048	1380	1	2742	1518	1	1518	886	4
1774	1489	1	1207	1609	1	2045	1383	1	2739	1508	1	1529	888	4
1796	1495	1	1190	1603	1	2017	1457	1	2737	1498	1	1671	896	4
1814	1501	1	1188	1597	1	2048	1460	1	2727	1489	1	1347	919	4
1830	1509	1	1177	1589	1	2094	1403	1	2721	1483	1	1310	893	4
1844	1516	1	1167	1577	1	2039	1394	1	2704	1474	1	1315	878	4
1851	1524	1	1167	1572	1	2294	1375	4	2123	1460	4	1355	1053	4
1859	1531	1	1201	1468	1	2012	1348	1	2714	1537	1	1365	1051	4
1866	1541	1	1205	1456	1	1997	1348	1	2135	1444	4	1377	1052	4
1874	1545	1	1213	1448	1	1984	1350	1	2737	1522	1	1452	1161	2
1889	1547	1	1219	1438	1	1974	1354	1	2143	1429	4	1449	1161	1
1918	1551	1	1229	1428	1	1971	1359	1	2739	1509	1	1442	1160	1
1967	1558	1	1239	1425	1	1937	1452	1	2159	1416	4	1437	1156	1
2005	1567	1	1251	1421	1	1939	1465	1	2734	1491	1	1434	1151	1
2050	1578	1	1272	1424	1	1943	1476	1	1375	1119	4	1434	1146	1
2075	1592	1	1446	1462	1	1949	1483	1	1374	1097	1	1433	1142	1
2093	1602	1	1190	1572	4	1961	1492	1	1457	884	4	1433	1137	1
2101	1611	1	1330	1599	1	1969	1492	1	1461	856	1	1436	1133	1
2107	1617	1	1348	1570	1	1978	1492	1	1444	855	1	1440	1129	1
2253	1642	1	1363	1548	1	1992	1494	1	1441	890	1	1445	1127	1
2290	1649	1	1377	1530	1	2821	1609	1	1457	885	1	1451	1129	1
2317	1655	1	1385	1521	1	2889	1620	1	1435	891	4	1456	1133	1
2358	1655	1	1403	1511	1	2941	1630	1	1438	860	1	1459	1137	1
2383	1655	1	1392	1516	2	2971	1636	1	1426	860	1	1461	1143	1
2413	1648	1	1780	1597	1	3010	1637	1	1423	893	1	1463	1149	1
2431	1645	1	1866	1618	1	3035	1637	1	1434	890	1	1460	1154	1
2488	1749	1	1908	1632	1	3065	1637	1	1416	866	4	1457	1157	1
2146	1700	2	1950	1646	1	3085	1633	1	1413	893	1	1454	1160	1
2392	1750	1	1988	1665	1	3092	1633	1	1400	894	1	1449	1161	1
2402	1774	1	2016	1683	1	3043	1630	2	1403	867	1	1413	1151	4
2515	1767	1	2037	1704	1	3262	1660	1	1418	864	1	1480	1158	4
2508	1747	1	2049	1717	1	3283	1665	1	1389	897	4	1490	1158	4
2390	1743	1	2053	1722	1	3304	1661	1	1392	883	1	1500	1149	4
2389	1751	1	2053	1728	1	3313	1658	1	1396	874	1	2137	1751	4
2503	1771	4	2051	1731	1	3310	1654	1	1399	866	1	2137	1764	1
2546	1848	1	2045	1735	1	3285	1593	1	1384	868	1	2145	1753	1
2552	1860	1	2037	1738	1	3269	1561	4	1379	874	1	2148	1767	1
2554	1873	1	2023	1740	1	3221	1480	1	1378	879	1	2172	1770	4
2548	1876	1	2004	1741	1	3218	1478	1	1380	888	1	2169	1772	1
2536	1878	1	1992	1741	1	3217	1474	1	1378	899	1	2159	1768	1
2526	1875	1	1978	1740	1	3209	1474	1	1392	896	1	2159	1757	1
2520	1865	1	1963	1741	1	3186	1475	1	1165	1190	4	2170	1757	1
2509	1852	1	1948	1738	1	3154	1481	1	1146	1195	4	2193	1776	4
2496	1835	1	1933	1736	1	3120	1481	1	1124	1196	4	2182	1776	1
2458	1773	1	1915	1733	1	3090	1483	1	1009	1190	4	2178	1762	1
2282	1659	4	1891	1726	1	3053	1483	1	610	1368	4	2190	1762	1
2333	1672	1	1333	1593	1	3019	1483	1	623	1370	4	2198	1772	4
2352	1681	1	1234	1582	4	2984	1481	1	645	1370	4	2206	1771	1
2371	1691	1	1234	1575	1	2942	1475	1	651	1370	4	2220	1786	4
2386	1694	1	1236	1565	1	2900	1470	1	665	1374	4	2214	1770	1
2406	1696	1	1265	1485	1	2891	1465	1	681	1375	4	2227	1786	4
2413	1696	1	1268	1482	1	2733	1452	1	704	1376	4	2238	1787	1
2441	1743	1	1276	1478	1	2114	1478	2	719	1375	4	2230	1771	1
2412	1697	4	1285	1474	1	2177	1386	1	736	1377	4	2250	1791	4
2452	1743	1	1290	1472	1	2118	1470	4	746	1377	4	2244	1774	1
2470	1772	4	1308	1474	1	2675	1542	1	765	1377	4	2255	1777	1
2523	1843	1	1329	1477	1	2710	1547	1	780	1375	4	2263	1794	1
2486	1771	1	1360	1482	1	2740	1548	1	678	1355	4	2251	1789	1
2472	1745	4	1833	1576	1	2778	1550	1	708	1357	4	2276	1800	4
2416	1652	1	1855	1592	4	2797	1550	1	1203	1378	4	2273	1778	1
2419	1647	1	1989	1614	1	2811	1550	1	1220	1378	4			

2·2 ヘリコプターの回転翼とSCREEN文

SCREEN文が上手に使えるようになると、いろいろアイデアがわいてきて困るぐらいでしょう。私もSCREEN文に慣れてしまうと、アイデアがたくさんでてきて、体力がついていかない情況に陥りました。私が何よりやってみたかったのはヘリコプターの回転翼を回してみることです。先ほど勉強しましたように、640×200の単色モードでは3枚の画面が使えます。前に手まりを回しました時にも述べましたが、一定方向への運動として人間が認知するには3枚の画面が必要なのです。

そこで回転翼が4枚翼の場合、60°ずつ回転した翼をもつ3枚の図があればよいことになります。口で言うのは簡単ですが、3枚の図面が、ずれない保証を与えるのはなかなか大変です。この難問を研究室の西山君が面白いテクニックで解決しました。図面は1枚で良く、他の2枚は回転翼をプログラム的に回転させるというやり方を作り出しました。これによって我々の作業は大変楽になりました。それでは次ページからの画面の仕上りをごらん下さい。

640×200モードの場合、

`SCREEN 1,,,1` とすると第1画面のヘリコプターが見えます。

`SCREEN 1,,,2` とすると第2画面のヘリコプターが見えます。回転翼は60°回転しています。

`SCREEN 1,,,4` とすると第3画面のヘリコプターが見えます。回転翼は120°回転しています。

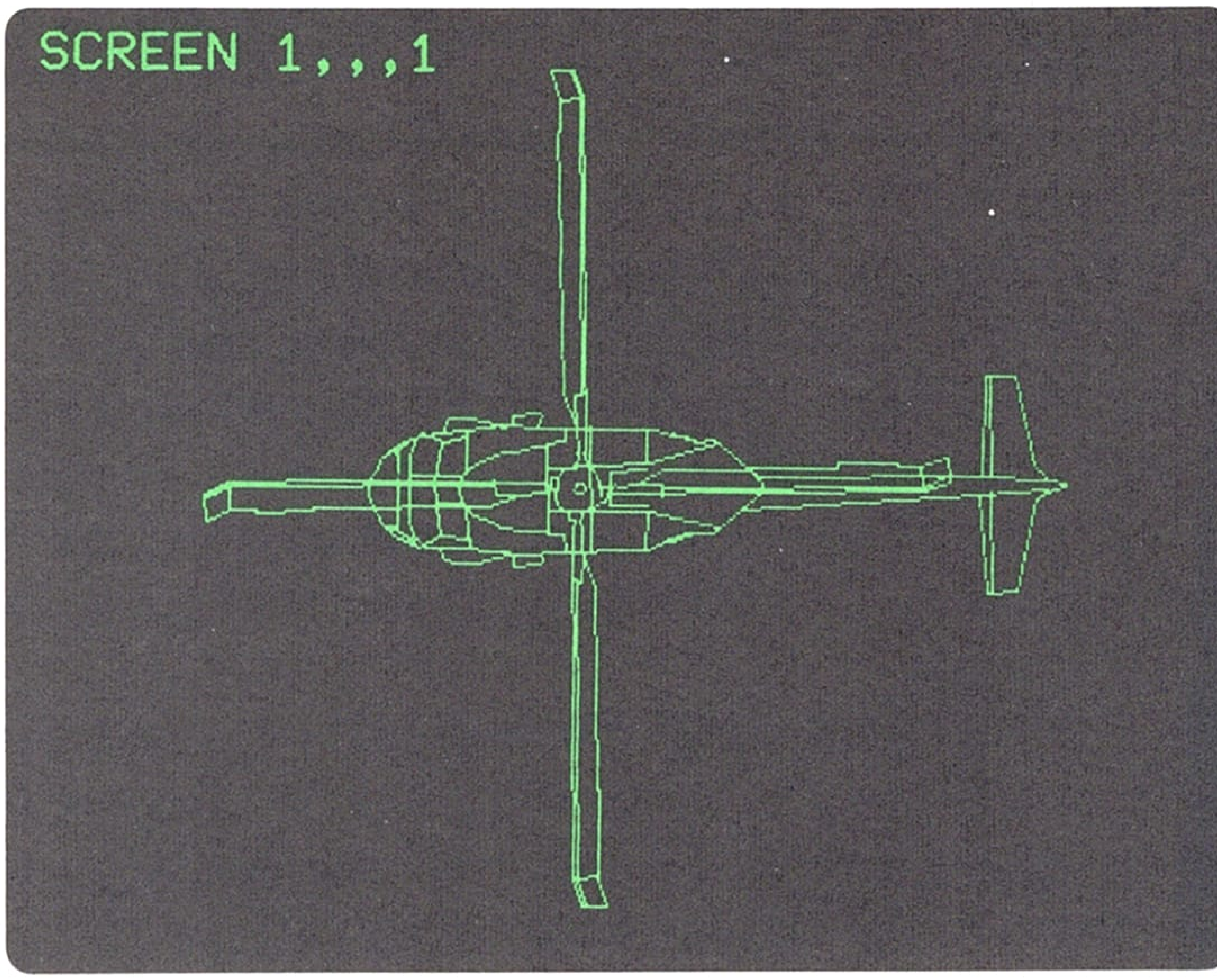
ついでに3画面全部を合成して見ておきましょう。それにはどうするのだったでしょうか。

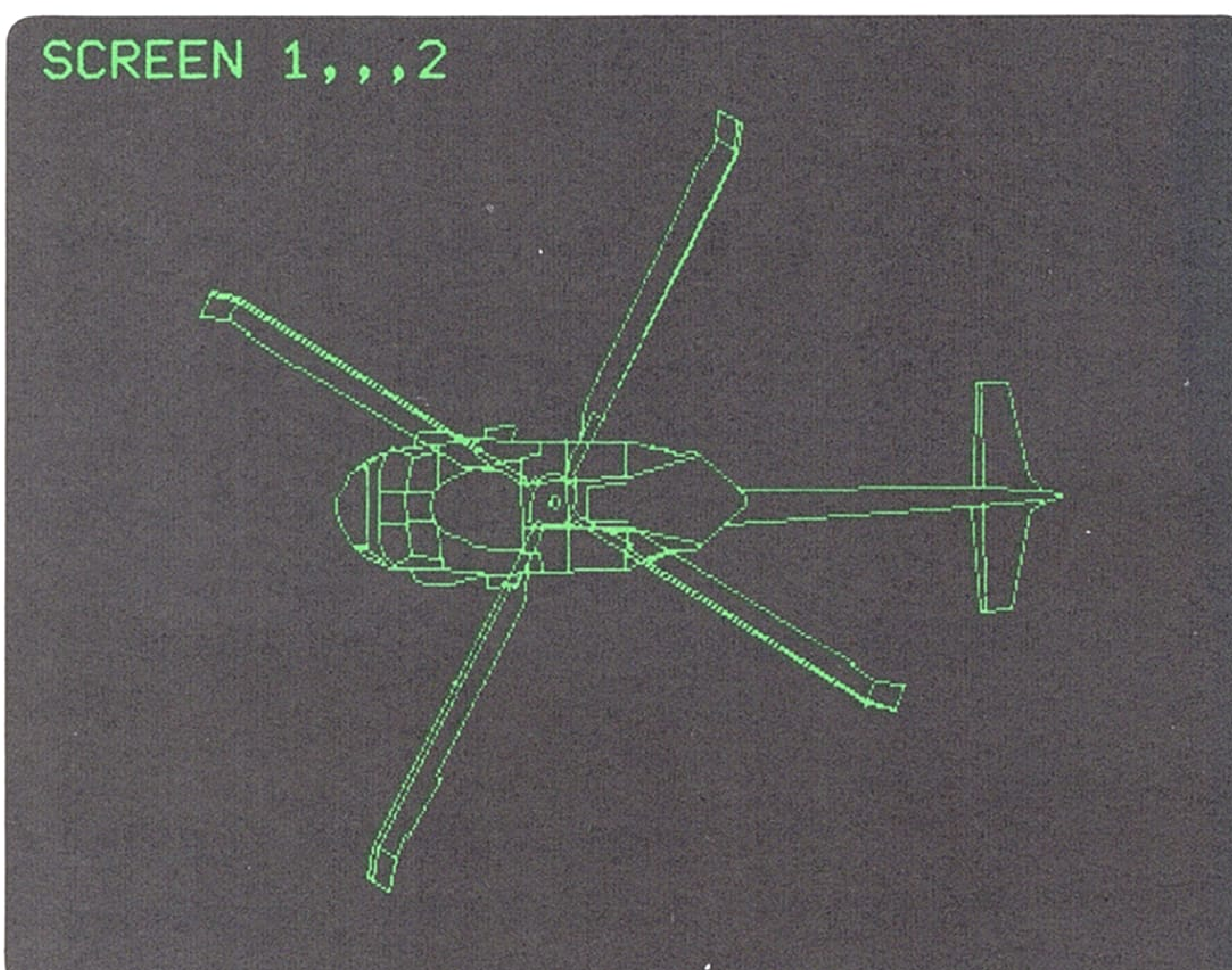
```
SCREEN 1,,,7
```

だったですね。ヘリコプターの胴体が、3画面を重ねても全くずれていないことに注意して下さい。3画面全部の合成では回転翼が回っていることの静止的説明にすぎません。本当に回転翼を回したいのです。それにはどうしたら良いでしょう。『画面を次々に切りかえるといいんじゃないか。』とおっしゃる方がいればそれで正解です。

原理的には、

```
SCREEN 1,,,1
SCREEN 1,,,2
SCREEN 1,,,4
```



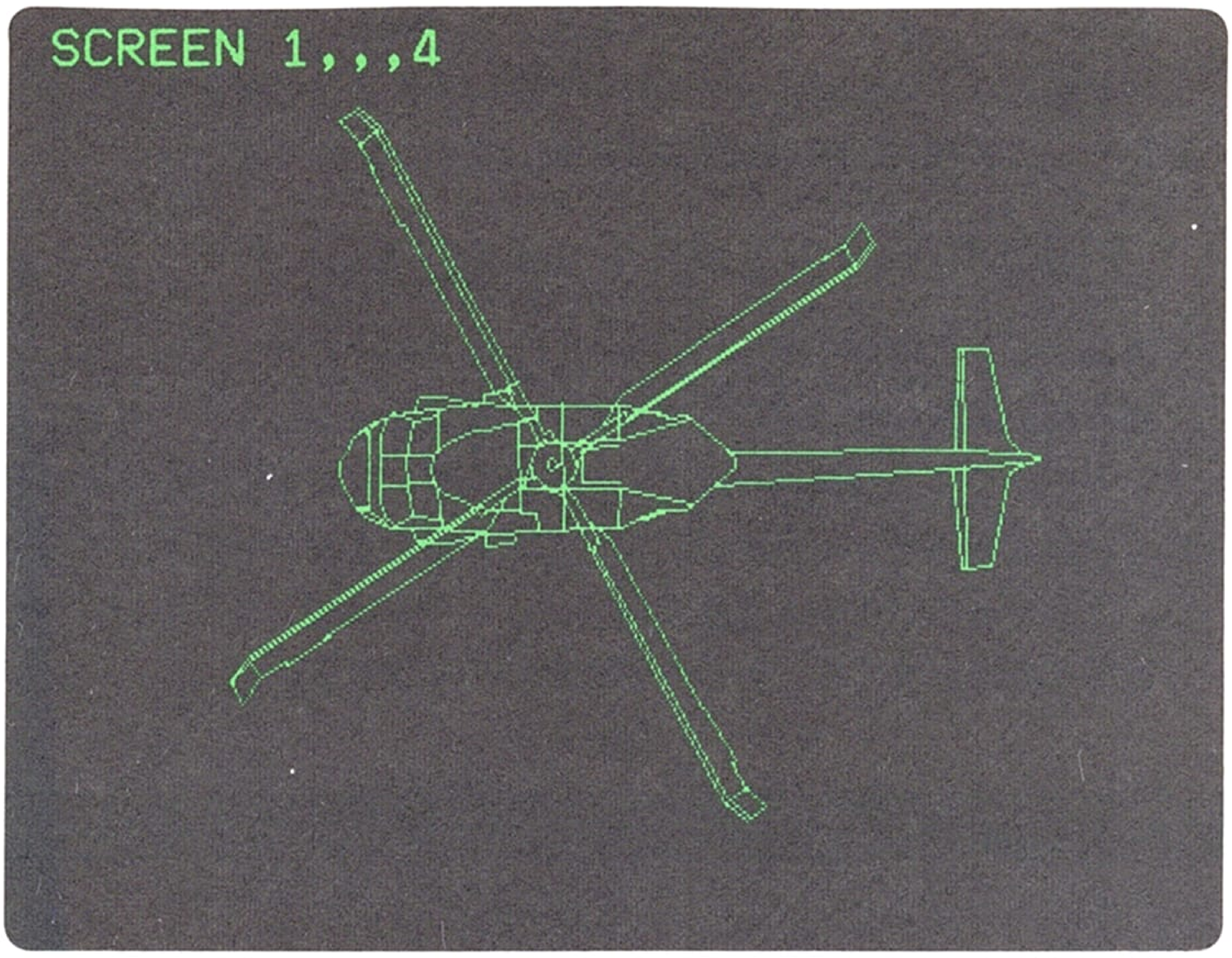


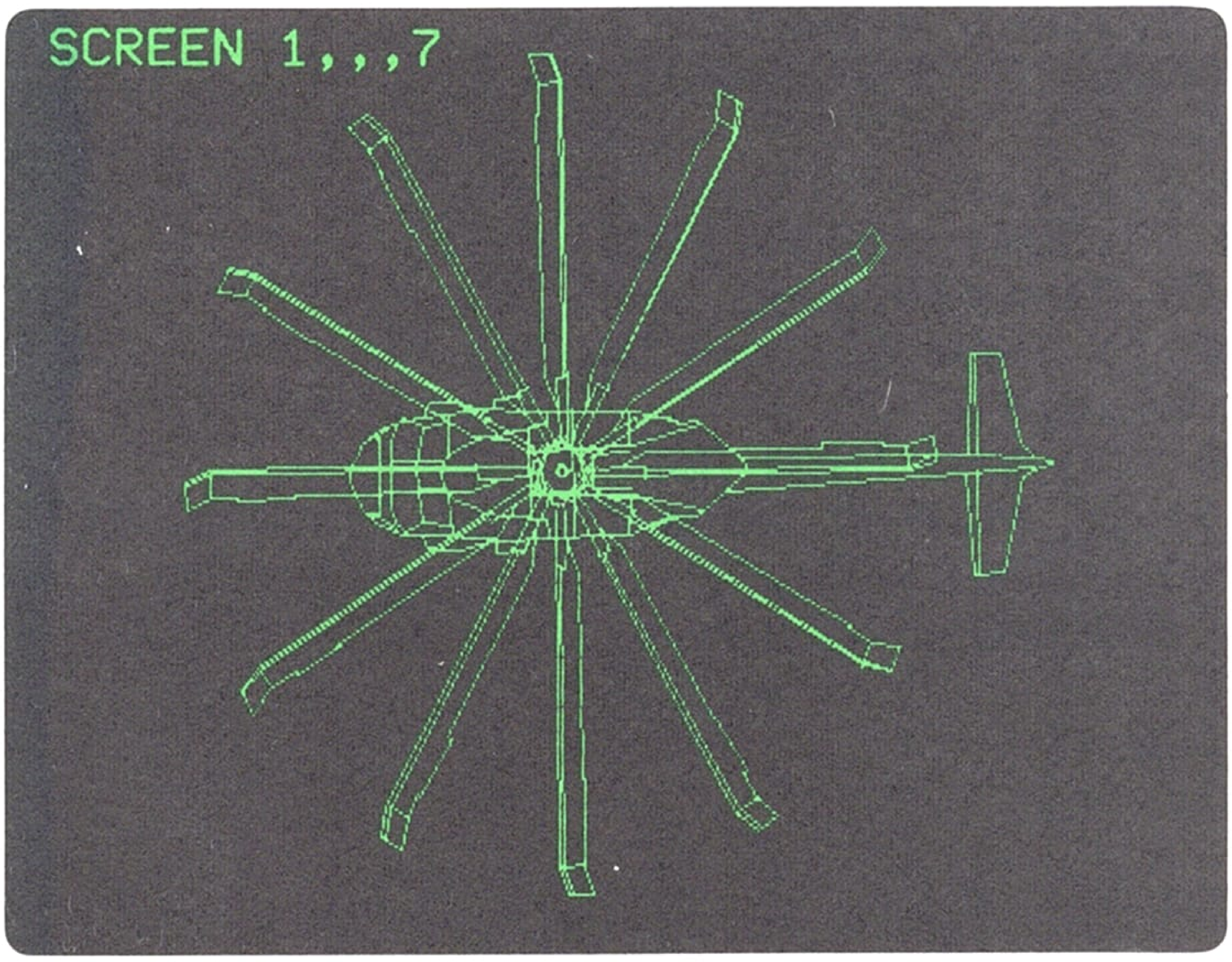
をループで回せば良いのですが、これは、次のように書いた方がスマートです。

```
110 FOR J=0 TO 2
120 SCREEN 1,,,2^J
130 NEXT J
```

2^Jは2のベキ乗をあらわしています。

110から130まででで1回だけ回転翼はまわりました。これを100回ま





わしたいなら、次のようにすればできます。

```
100 FOR I=1 TO 100
110 FOR J=0 TO 2
120 SCREEN 1,,,2^J
130 NEXT J
140 NEXT I
```

実際にやってみますと、これでは速すぎて回転はするのですが、ブーンとまわっている感じがでないのです。タイミングをとることが必要です。行番号340のSCREEN文の後にタイマーとして、

```
FOR T=1 TO 100 : NEXT T
```

を入れておくと本当に回っているという感じがでてきます。

我々が表示用に使っているプログラムをお見せします。これもまたフロッピーディスク装置からデータを読み出していますので、わかりにくい点があるでしょうが、SCREEN文の使い方だけを注目して下さい。どうやっているか、思想だけはしっかりと理解して下さい。

思想を理解しますと、わかりが大変早くなります。

●サンプル・プログラム●

```
100 WIDTH 80,25
110 CONSOLE 0,25,0,1
120 SCREEN 1,3:CLS 3:SCREEN 1,0
140 OPEN "GRPDAT3" FOR INPUT AS #1
150 FOR I=0 TO 2
160 SCREEN 1,0,I,1
170 WINDOW (0,350)-(1800,1700)
    :VIEW (0,0)-(639,199)
180  IF EOF(1) THEN GOTO *FINISH
190  INPUT #1,X,Y,F
200  IF X=0 AND Y=0 THEN 300
210  IF F<>1 OR PCNT=0 THEN PSET(X,Y):GOTO 230
220  LINE(XOLD,YOLD)-(X,Y)
230  XOLD=X:YOLD=Y
240  PCNT=1
250  GOTO 180
300 PCNT=0
310 NEXT I
320  *FINISH
325 CLOSE
330 FOR I=1 TO 100
335 FOR J=0 TO 2
340  SCREEN ,,,2^J
350  FOR T=1 TO 100:NEXT T ←―― タイマー
360 NEXT J
370 NEXT I
```

2·3 ヘリコプターをDATA文で描く

ヘリコプターの場合は、3画面に胴体を同じデータで描いています。従ってデータに重複があります。我々のやり方ではフロッピーディスク装置を使っていますので、重複を気にせずに入力してあります。

DATA文に直す際に重複を除こうかとも考えましたが、原データに手を加えますと、とかく間違いが発生するので、そのままにしてあります。

READ～DATA文を使ったプログラムを下に示しておきます。

DATA文の作り方は前と良く似ていますが、＊＊＊……＊＊＊と続く部分になったら0,0,0のデータを書いて下さい。これは一画面の書き込み終了のマークです。これを2回くり返し、すべての入力が終了したら0,0,9を書いて下さい。これで全てのデータができ上ります。データの構成は次のようになっています。

```
DATA （ 第1画面のデータ ）,0,0,0
DATA （ 第2画面のデータ ）,0,0,0
DATA （ 第3画面のデータ ）,0,0,9
```

ヘリコプターの翼がまわり出すと、ちょっとしたものですが、少々骨がおれると思います。

●ヘリコプター・プログラムリスト●

```
100 WIDTH 80,25:CONSOLE 0,25,0,1
110 SCREEN 0,3:CLS 3:SCREEN 0,0
120 '
130 WINDOW (0,400)-(1800,1700):VIEW(0,0)-(639,199)
140 FOR I=0 TO 2
150 SCREEN 1,0,I,7
160  READ X,Y,F
170  IF F=9 THEN GOTO *FINISH
180  IF X=0 AND Y=0 THEN 240
190  IF F<>1 OR PCNT=0 THEN PSET(X,Y):GOTO 210
200  LINE(XOLD,YOLD)-(X,Y)
210  XOLD=X:YOLD=Y
220  PCNT=1
230  GOTO 160
240 PCNT=0
250 NEXT I
260 '
270  *FINISH
280 CLOSE
290 FOR I=1 TO 100
300 FOR J=0 TO 2
```

```
310   SCREEN ,,,2^J
320   FOR T=1 TO 100:NEXT T
330 NEXT J
340 NEXT I
350 '
360 DATA 1502,948,1,1502,945,1,1479,941,1,1462,926,1,0,0,0
370 DATA 1502,948,1,1502,945,1,1479,941,1,1462,926,1,0,0,0
380 DATA 1502,948,1,1502,945,1,1479,941,1,1462,926,1,0,0,9
```

●ヘリコプターを描くデータ●

1502	948	1	1478	953	1	758	901	2	1042	1006	1	869	963	1
1502	945	1	1398	939	2	729	909	1	983	976	2	869	959	1
1479	941	1	1479	941	1	700	899	1	934	976	1	860	959	1
1462	926	1	1098	981	2	726	1013	2	934	938	1	855	961	1
1438	828	1	1125	959	1	720	980	1	981	936	1	850	963	1
1401	826	1	1125	949	1	719	949	1	1412	1067	2	822	965	1
1397	939	1	1115	943	1	727	910	1	1409	957	1	787	978	1
1118	943	1	812	1032	2	687	978	2	1409	938	2	569	983	1
1065	901	1	795	1038	1	756	977	1	1407	829	1	569	985	1
1039	904	1	773	1044	1	688	945	2	818	1033	2	498	990	1
1032	898	1	747	1045	1	753	946	1	818	1046	1	498	985	1
1011	897	1	732	1040	1	700	1021	2	851	1044	1	467	986	1
988	891	1	728	1028	1	696	1018	1	854	1038	1	437	998	1
981	891	1	726	899	2	689	1008	1	850	1033	1	433	969	1
851	892	1	729	892	1	687	994	1	815	887	2	465	955	1
847	890	1	735	882	1	686	980	1	815	877	1	605	952	1
807	889	1	746	880	1	686	947	1	847	874	1	856	951	1
726	898	1	760	880	1	686	931	1	850	878	1	856	951	1
701	901	1	779	883	1	686	921	1	845	887	1	861	954	1
680	907	1	793	887	1	690	915	1	1356	918	2	862	947	1
665	918	1	805	891	1	692	909	1	1357	947	1	870	945	1
654	926	1	762	1025	2	698	900	1	1325	957	1	872	942	1
644	941	1	759	982	1	693	1020	2	931	964	1	876	939	1
638	951	1	758	940	2	673	1005	1	929	973	1	882	936	1
638	968	1	758	896	1	663	1006	1	923	973	1	893	934	1
644	986	1	883	1028	2	688	904	2	917	977	1	897	920	1
653	1000	1	879	1009	1	673	920	1	912	981	1	894	918	1
667	1012	1	860	1011	1	662	919	1	905	983	1	896	914	1
685	1021	1	860	1006	1	658	1003	2	900	985	1	893	889	1
702	1025	1	842	1008	1	674	1002	1	899	997	1	875	855	1
723	1027	1	825	1006	1	673	921	1	902	1001	1	869	621	1
744	1027	1	803	1003	1	661	919	1	901	1005	1	867	623	1
766	1028	1	780	994	1	860	1006	2	902	1030	1	866	563	1
783	1030	1	767	990	1	857	912	1	918	1067	1	869	563	1
792	1030	1	756	981	1	912	1026	2	925	1293	1	869	529	1
811	1030	1	752	971	1	910	888	1	927	1293	1	856	496	1
817	1033	1	750	958	1	859	983	2	927	1362	1	885	494	1
850	1033	1	753	946	1	983	983	1	926	1362	1	896	526	1
855	1030	1	761	940	1	1074	998	1	926	1391	1	903	912	1
983	1026	1	770	933	1	858	934	2	939	1421	1	905	917	1
1011	1014	1	786	925	1	983	929	1	907	1424	1	902	918	1
1015	1018	1	804	919	1	1071	906	1	897	1393	1	903	925	1
1042	1008	1	818	917	1	980	1025	2	891	1003	1	907	929	1
1064	1008	1	842	912	1	982	984	1	890	999	1	910	933	1
1099	980	1	847	912	1	981	929	2	893	997	1	915	940	1
1395	958	1	857	912	1	979	890	1	893	993	1	918	945	1
1407	1071	1	857	909	1	1012	896	2	885	992	1	920	948	1
1442	1066	1	878	906	1	1032	903	1	885	983	1	922	951	1
1466	964	1	879	890	1	1040	902	1	879	983	1	922	955	1
1479	955	1	760	1015	2	1011	1015	2	873	978	1	937	955	1
1501	948	1	727	1013	1	1008	1017	2	870	973	1	940	953	1
1395	958	2	705	1024	1	1032	1008	1	869	968	1	944	954	1

968	951	1
1003	934	1
1216	931	1
1218	926	1
1292	926	1
1292	930	1
1322	928	1
1355	918	1
1321	927	2
1323	954	1
926	1390	2
898	1391	1
464	987	2
464	958	1
869	530	2
897	526	1
968	951	2
999	951	1
999	966	1
903	1034	2
901	1063	1
891	1061	1
823	965	2
793	965	1
791	949	1
893	887	2
890	853	1
903	852	1
882	494	2
892	527	1
898	853	1
1356	940	2
1325	952	1
1000	957	1
914	1423	2
902	1393	1
895	1064	1
437	973	2
467	960	1
792	954	1
893	964	2
891	958	1
893	952	1
896	951	1
899	951	1
901	953	1
903	955	1
903	959	1
901	961	1
897	961	1
894	961	1
924	965	2
924	955	1
900	984	2
884	983	1
870	959	2
871	945	1
893	931	2
908	933	1
***************		
1502	948	1
1502	945	1
1479	941	1
1462	926	1
1438	828	1
1401	826	1
1397	939	1
1118	943	1
1065	901	1
1039	904	1
1032	898	1

1011	897	1
988	891	1
981	891	1
851	892	1
847	890	1
807	889	1
726	898	1
701	901	1
680	907	1
665	918	1
654	926	1
644	941	1
638	951	1
638	968	1
644	986	1
653	1000	1
667	1012	1
685	1021	1
702	1025	1
723	1027	1
744	1027	1
766	1028	1
783	1030	1
792	1030	1
811	1030	1
817	1033	1
850	1033	1
855	1030	1
983	1026	1
1011	1014	1
1015	1018	1
1042	1008	1
1064	1008	1
1099	980	1
1395	958	1
1407	1071	1
1442	1066	1
1466	964	1
1479	955	1
1501	948	1
1395	958	2
1478	953	1
1398	939	2
1479	941	1
1098	981	2
1125	959	1
1125	949	1
1115	943	1
812	1032	2
795	1038	1
773	1044	1
747	1045	1
732	1040	1
728	1028	1
726	899	2
729	892	1
735	882	1
746	880	1
760	880	1
779	883	1
793	887	1
805	891	1
762	1025	2
759	982	1
758	940	2
758	896	1
883	1028	2
879	1009	1
860	1011	1
860	1006	1
842	1008	1

825	1006	1
803	1003	1
780	994	1
767	990	1
756	981	1
752	971	1
750	958	1
753	946	1
761	940	1
770	933	1
786	925	1
804	919	1
818	917	1
842	912	1
847	912	1
857	912	1
857	909	1
878	906	1
879	890	1
760	1015	2
727	1013	1
705	1024	1
758	901	2
729	909	1
700	899	1
726	1013	2
720	980	1
719	949	1
727	910	1
687	978	2
756	977	1
688	945	2
753	946	1
700	1021	2
696	1018	1
689	1008	1
687	994	1
686	980	1
686	947	1
686	931	1
686	921	1
690	915	1
692	909	1
698	900	1
693	1020	2
673	1005	1
663	1006	1
688	904	2
673	920	1
662	919	1
658	1003	2
674	1002	1
673	921	1
661	919	1
860	1006	2
857	912	1
912	1026	2
910	888	1
859	983	2
983	983	1
1074	998	1
858	934	2
983	929	1
1071	906	1
980	1025	2
982	984	1
981	929	2
979	890	1
1012	896	2
1032	903	1
1040	902	1

1011	1015	2
1008	1017	2
1032	1008	1
1042	1006	1
983	976	2
934	976	1
934	938	1
981	936	1
1412	1067	2
1409	957	1
1409	938	2
1407	829	1
818	1033	2
818	1046	1
851	1044	1
854	1038	1
850	1033	1
815	887	2
815	877	1
847	874	1
850	878	1
845	887	1
700	1373	2
675	1359	1
682	1326	1
873	981	1
866	975	1
869	970	1
869	963	1
868	956	1
870	949	1
870	944	1
860	937	1
856	938	1
853	935	1
830	923	1
790	919	1
591	812	1
590	813	1
530	779	1
531	778	1
506	764	1
473	760	1
487	731	1
519	738	1
859	927	1
863	928	1
863	932	1
867	934	1
872	928	1
880	932	1
883	927	1
890	924	1
896	924	1
901	926	1
905	928	1
908	930	1
913	922	1
914	917	1
914	912	1
927	887	1
933	850	1
1038	658	1
1036	657	1
1067	593	1
1071	596	1
1086	569	1
1091	537	1
1118	548	1
1114	582	1
1046	705	1

922	923	1
922	923	1
917	926	1
922	930	1
920	938	1
922	941	1
922	946	1
922	953	1
918	964	1
928	974	1
931	972	1
934	976	1
957	986	1
995	987	1
1201	1099	1
1200	1097	1
1253	1126	1
1251	1128	1
1281	1145	1
1316	1150	1
1303	1177	1
1270	1170	1
932	983	1
927	982	1
927	979	1
921	977	1
915	978	1
910	979	1
902	980	1
896	980	1
892	980	1
889	980	1
885	978	1
878	991	1
878	995	1
875	998	1
866	1020	1
863	1059	1
759	1245	1
762	1249	1
725	1313	1
722	1311	1
709	1338	1
701	1372	1
710	1338	2
686	1326	1
507	764	2
520	739	1
1087	566	2
1112	580	1
1280	1145	2
1269	1171	1
866	1020	2
850	1047	1
837	1039	1
826	922	2
802	906	1
809	898	1
926	887	2
941	861	1
956	868	1
959	987	2
990	1001	1
984	1013	1
1305	1174	2
1271	1166	1
986	1008	1
681	1362	2
686	1329	1
845	1045	1
484	737	2

516	742	1
804	900	1
1112	549	2
1109	582	1
951	866	1
892	949	2
898	950	1
902	955	1
902	958	1
900	960	1
898	961	1
895	962	1
891	960	1
891	957	1
893	954	1
894	951	1
876	975	2
884	980	1
871	945	2
880	931	1
908	931	2
920	939	1
921	965	2
911	977	1
***************		
1502	948	1
1502	945	1
1479	941	1
1462	926	1
1438	828	1
1401	826	1
1397	939	1
1118	943	1
1065	901	1
1039	904	1
1032	898	1
1011	897	1
988	891	1
981	891	1
851	892	1
847	890	1
807	889	1
726	898	1
701	901	1
680	907	1
665	918	1
654	926	1
644	941	1
638	951	1
638	968	1
644	986	1
653	1000	1
667	1012	1
685	1021	1
702	1025	1
723	1027	1
744	1027	1
766	1028	1
783	1030	1
792	1030	1
811	1030	1
817	1033	1
850	1033	1
855	1030	1
983	1026	1
1011	1014	1
1015	1018	1
1042	1008	1
1064	1008	1
1099	980	1
1395	958	1

1407	1071	1
1442	1066	1
1466	964	1
1479	955	1
1501	948	1
1395	958	2
1478	953	1
1398	939	2
1479	941	1
1098	981	2
1125	959	1
1125	949	1
1115	943	1
812	1032	2
795	1038	1
773	1044	1
747	1045	1
732	1040	1
728	1028	1
726	899	2
729	892	1
735	882	1
746	880	1
760	880	1
779	883	1
793	887	1
805	891	1
762	1025	2
759	982	1
758	940	2
758	896	1
883	1028	2
879	1009	1
860	1011	1
860	1006	1
842	1008	1
825	1006	1
803	1003	1
780	994	1
767	990	1
756	981	1
752	971	1
750	958	1
753	946	1
761	940	1
770	933	1
786	925	1
804	919	1
818	917	1
842	912	1
847	912	1
857	912	1
857	909	1
878	906	1
879	890	1
760	1015	2
727	1013	1
705	1024	1
758	901	2
729	909	1
700	899	1
726	1013	2
720	980	1
719	949	1
727	910	1
687	978	2
756	977	1
688	945	2
753	946	1
700	1021	2
696	1018	1

689	1008	1
687	994	1
686	980	1
686	947	1
686	931	1
686	921	1
690	915	1
692	909	1
698	900	1
693	1020	2
673	1005	1
663	1006	1
688	904	2
673	920	1
662	919	1
658	1003	2
674	1002	1
673	921	1
661	919	1
860	1006	2
857	912	1
912	1026	2
910	888	1
859	983	2
983	983	1
1074	998	1
858	934	2
983	929	1
1071	906	1
980	1025	2
982	984	1
981	929	2
979	890	1
1012	896	2
1032	903	1
1040	902	1
1011	1015	2
1008	1017	2
1032	1008	1
1042	1006	1
983	976	2
934	976	1
934	938	1
981	936	1
1412	1067	2
1409	957	1
1409	938	2
1407	829	1
818	1033	2
818	1046	1
851	1044	1
854	1038	1
850	1033	1
815	887	2
815	877	1
847	874	1
850	878	1
845	887	1
635	577	2
659	562	1
684	585	1
887	923	1
896	920	1
899	925	1
905	928	1
911	931	1
916	936	1
921	939	1
932	934	1
933	929	1
937	928	1

959	915	1
983	882	1
1175	763	1
1174	762	1
1234	727	1
1234	728	1
1259	713	1
1279	687	1
1297	713	1
1275	737	1
941	938	1
938	941	1
935	939	1
931	941	1
934	948	1
926	953	1
929	958	1
928	966	1
925	971	1
921	974	1
917	977	1
914	979	1
918	987	1
922	990	1
927	993	1
942	1016	1
971	1040	1
1084	1227	1
1086	1226	1
1126	1285	1
1122	1287	1
1138	1313	1
1163	1333	1
1140	1351	1
1112	1331	1
1040	1211	1
913	994	1
913	994	1
913	988	1
907	991	1
901	985	1
897	985	1
893	983	1
887	979	1
880	970	1
866	974	1
866	978	1
861	978	1
841	993	1
821	1026	1
621	1148	1
624	1148	1
572	1179	1
571	1177	1
541	1194	1
519	1222	1
503	1197	1
525	1172	1
856	973	1
859	969	1
862	971	1
867	966	1
869	961	1
871	956	1
874	948	1
877	943	1
879	940	1
880	937	1
884	935	1
876	922	1
873	920	1

872	916	1
857	897	1
825	875	1
716	692	1
711	693	1
674	629	1
677	627	1
660	602	1
635	578	1
660	603	2
682	588	1
1258	714	2
1273	738	1
1140	1315	2
1115	1330	1
542	1193	2
525	1171	1
857	897	2
842	870	1
855	863	1
962	912	2
988	899	1
991	909	1
942	1016	2
957	1042	1
944	1051	1
839	994	2
811	1014	1
804	1003	1
504	1200	2
528	1175	1
807	1007	1
654	566	2
680	587	1
846	866	1
1293	708	2
1273	733	1
992	904	1
1142	1346	2
1115	1326	1
948	1048	1
906	955	2
902	960	1
896	961	1
893	959	1
892	957	1
892	954	1
893	951	1
897	949	1
899	950	1
901	953	1
903	956	1
891	928	2
883	933	1
920	939	2
927	954	1
913	978	2
900	984	1
877	972	2
872	958	1
***************		

2·4 可変後退翼ジェット機を作る

スクリーン文を使って、もう少し面白いグラフィックスを楽しんでみましょう。ここでは可変後退翼のジェット機を作ってみましょう。640ドット×200ドットの単色モードを使うことにします。

基本的な配置案として次のように考えてみました。

SCREEN 1,,,1	第1画面──→ジェット機の胴体
SCREEN 1,,,2	第2画面──→後退翼を広げたもの
SCREEN 1,,,4	第3画面──→後退翼を後方へ下げたもの

このように配置すると、SCREEN文によって画面には次のものが見えるようになります（一部重複させます）。

SCREEN 1,,,1	第1画面──→ジェット機の胴体
SCREEN 1,,,2	第2画面──→広げられた後退翼
SCREEN 1,,,3	第1画面＋第2画面──→胴体＋広げられた後退翼
SCREEN 1,,,4	第3画面──→後方に下がった後退翼
SCREEN 1,,,5	第1画面＋第3画面──→胴体＋後方へ下がった後退翼
SCREEN 1,,,6	第2画面＋第3画面 ──→広げられた後退翼＋後方へ下がった後退翼
SCREEN 1,,,7	第1画面＋第2画面＋第3画面 ──→胴体＋広げられた後退翼＋後方へ下がった後退翼

7つのパラメータによっていろいろな場面をみることができます。我々にとっては後退翼を動かすことだけが必要なわけですから、次のようにすれば後退翼が動きます。

```
100 FOR I=1 TO 20
110 SCREEN 1,,,3
120 SCREEN 1,,,5
130 NEXT I
140 '
150 END
```

しかし、これでは速すぎて眼に止まりません。そこでタイミングルーチンをつけてやることにしましょう。これで完成です。プログラムを次ページに示します。

```
100 FOR I=1 TO 20
110 SCREEN 1,,,3
120  FOR J=1 TO 1000:NEXT J ←――――― タイマー
130 SCREEN 1,,,5
140  FOR J=1 TO 1000:NEXT J ←――――― タイマー
150 NEXT I
160 '
170 END
```

●可変後退翼ジェット機プログラム・リスト●

```
100 FOR I=1 TO 7:COLOR=(I,I):NEXT I
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 SCREEN 0,3:CLS 3:SCREEN 0,0
130 '
140 POIT=0
150 FOR LOOP=0 TO 2
160 SCREEN 1,,LOOP,7
170 WINDOW (200,800)-(1550,1700)
   :VIEW(0,0)-(639,199)
180 READ X,Y,F
190 IF X=0 AND Y=0 THEN GOTO 250
200 IF F<>1 OR POIT=0 THEN PSET(X,Y):GOTO 220   ― データの書き込み
210 LINE(XOLD,YOLD)-(X,Y)
220 XOLD=X:YOLD=Y
230 POIT=1
240 GOTO 180
250 POIT=0
260 NEXT LOOP
270 '
280 FOR I=1 TO 20
290 SCREEN 1,,,3
300  FOR J=1 TO 1000:NEXT J
310 SCREEN 1,,,5                ― 可変後退翼が開いたり閉じたりする
320  FOR J=1 TO 1000:NEXT J
330 NEXT I
340 '
350 END
360 '
370 DATA 448,1249,1,487,1248,1,548,1268,1,0,0,0
380 DATA 1007,1371,2,1074,1620,2,1082,1626,2
390 DATA 1008,1371,2,1073,1621,1,0,0,0
400 DATA 1023,1130,1,1284,1041,1,1292,1045,1,0,0,0
```

DATA文は未完成です。次ページの説明を読んで入力してください。

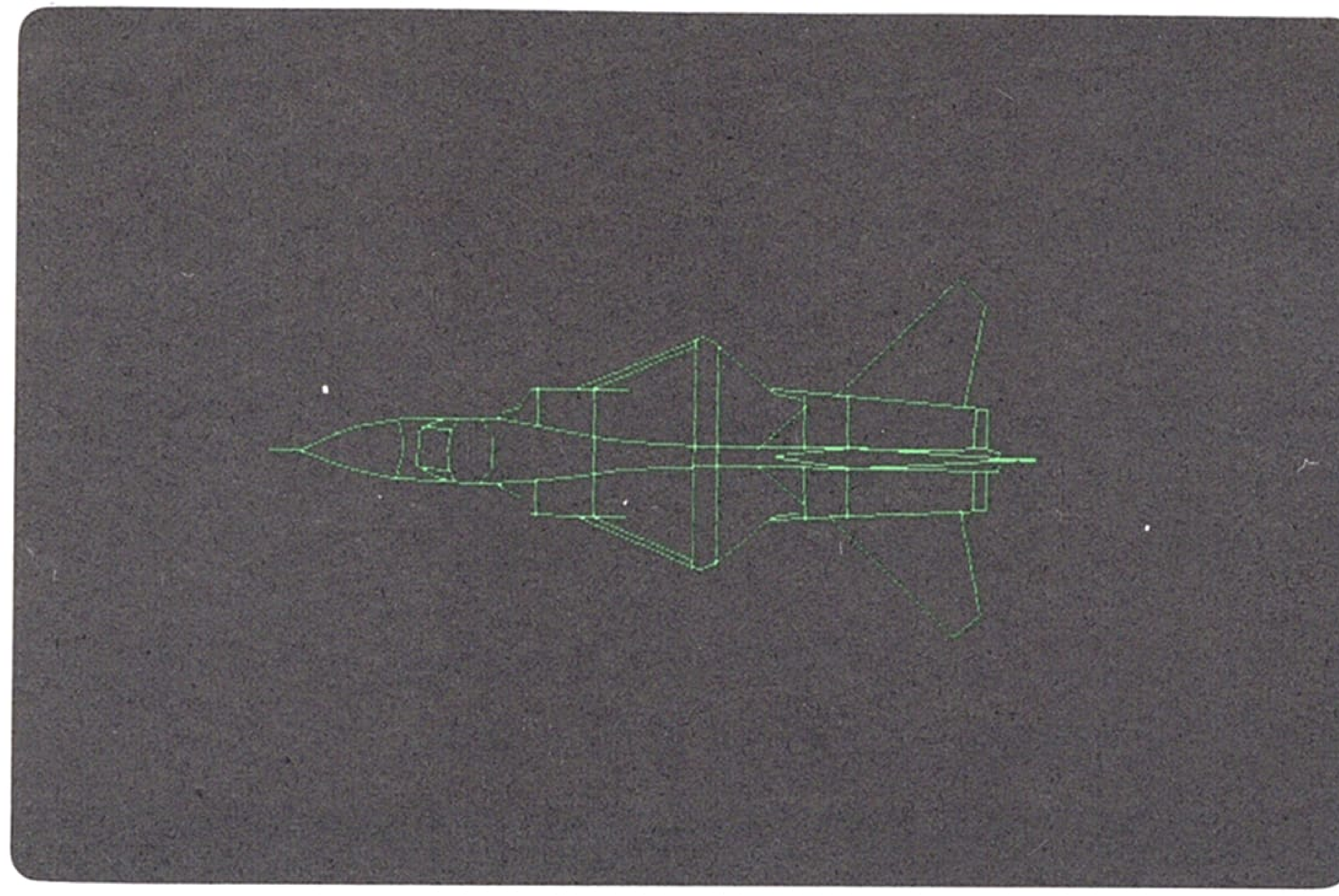
可変後退翼ジェット機の場合はデータ量が少ないので、挑戦してみて下さい。この場合もREAD～DATA文を使います。＊＊＊……＊のくるところがデータの切れ目です。ここにきたら,0,0,0を書いて下さい。3枚の画面のデータの終りには必ず0,0,0を書いて下さい。つまり次のようにしていただきたいと思います。

```
DATA ( 第1画面のデータ ),0,0,0
DATA ( 第2画面のデータ ),0,0,0
DATA ( 第3画面のデータ ),0,0,0
```

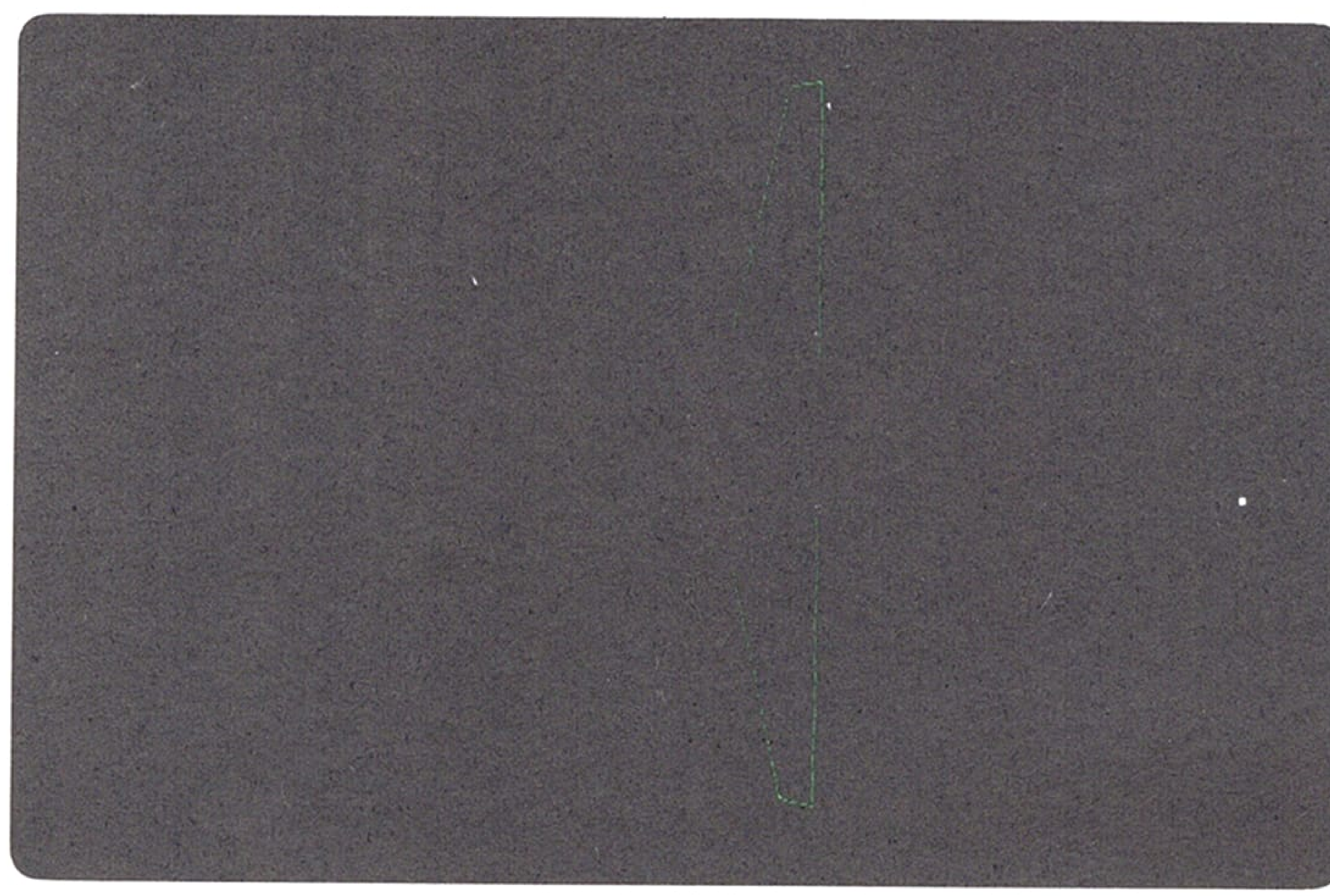
プログラムがうまく入力できますと可変後退翼がバタバタ開いたり閉じたりするのを見ることができます。是非試して下さい。

●可変後退翼プログラムのデータ●

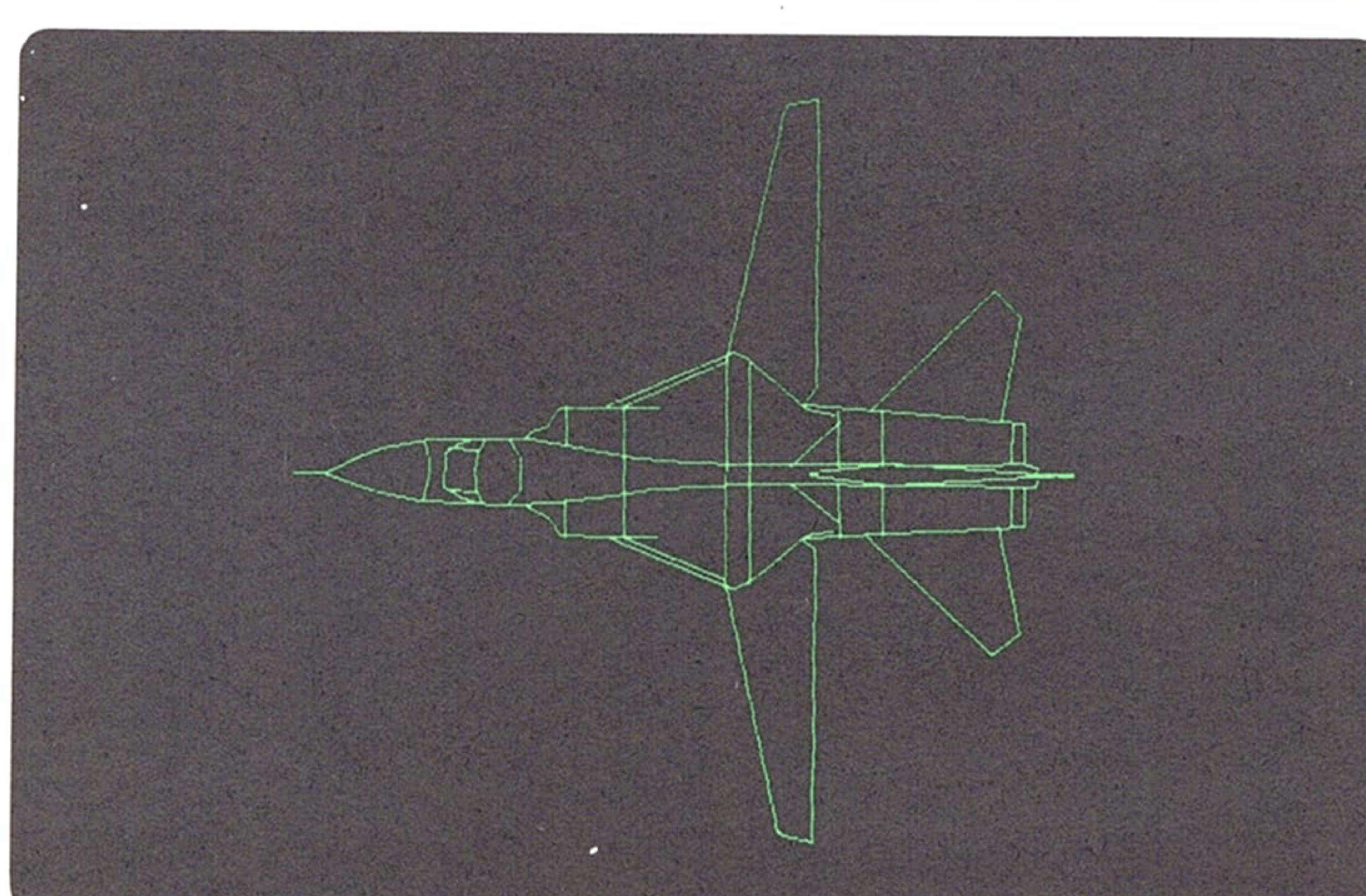
448	1249	1	1005	1369	1	696	1216	1	1037	1135	1
487	1248	1	1018	1370	1	686	1281	2	1153	1318	2
548	1268	1	1036	1363	1	666	1276	1	1151	1262	1
612	1279	1	1078	1337	1	641	1265	1	1150	1245	2
723	1282	1	1105	1319	1	643	1228	1	1150	1187	1
786	1280	1	1116	1315	1	662	1219	1	1207	1189	2
860	1273	1	1144	1311	1	682	1215	1	1207	1245	1
911	1267	1	1108	1319	2	646	1268	2	1207	1263	2
978	1264	1	1209	1315	1	685	1270	1	1207	1316	1
1035	1262	1	1353	1309	1	647	1225	2	1090	1263	2
1293	1264	1	1391	1307	1	688	1226	1	1152	1302	1
1389	1267	1	1392	1268	1	1186	1317	2	1089	1245	2
1394	1266	1	1146	1192	2	1350	1440	1	1152	1200	1
1407	1266	1	1111	1185	1	1385	1417	1	****************		
1408	1260	1	1099	1178	1	1357	1308	1	1007	1371	2
1407	1251	2	1037	1137	1	1195	1186	2	1074	1620	2
1409	1246	1	1029	1132	1	1353	1065	1	1082	1626	2
1394	1244	1	1016	1128	1	1387	1093	1	1008	1371	2
1391	1242	1	1003	1132	1	1363	1198	1	1073	1621	1
1392	1202	1	852	1182	1	1376	1202	2	1082	1626	1
1235	1190	1	920	1183	2	1374	1242	1	1117	1632	1
1108	1182	1	791	1184	1	1375	1268	2	1121	1331	1
1391	1242	2	782	1195	1	1375	1307	1	1106	1320	1
1335	1245	1	775	1203	1	1455	1254	2	1010	1127	2
1112	1243	1	757	1207	1	1215	1264	1	1081	879	1
1028	1242	1	746	1216	1	1115	1254	1	1087	870	1
959	1239	1	617	1217	2	1115	1250	1	1092	869	1
891	1235	1	621	1226	1	1215	1243	1	1126	866	1
815	1224	1	622	1238	1	1455	1255	1	1124	1163	1
753	1215	1	622	1250	1	797	1316	2	1106	1181	1
709	1215	1	620	1263	1	797	1279	1	****************		
625	1216	1	617	1272	1	799	1222	2	1023	1130	1
561	1225	1	614	1279	1	799	1182	1	1284	1041	1
531	1232	1	724	1282	2	873	1316	2	1292	1045	1
489	1248	1	740	1267	1	873	1181	1	1322	1068	1
747	1282	2	740	1234	1	872	1316	2	1146	1185	1
755	1287	1	727	1216	1	1005	1362	1	1010	1372	2
777	1296	1	693	1280	2	876	1181	2	1276	1465	1
791	1314	1	684	1269	1	1007	1137	1	1283	1465	1
852	1317	1	684	1259	1	1007	1129	2	1316	1439	1
917	1316	1	681	1247	1	1005	1371	1	1139	1319	1
852	1317	2	686	1229	1	1036	1366	2	****************		



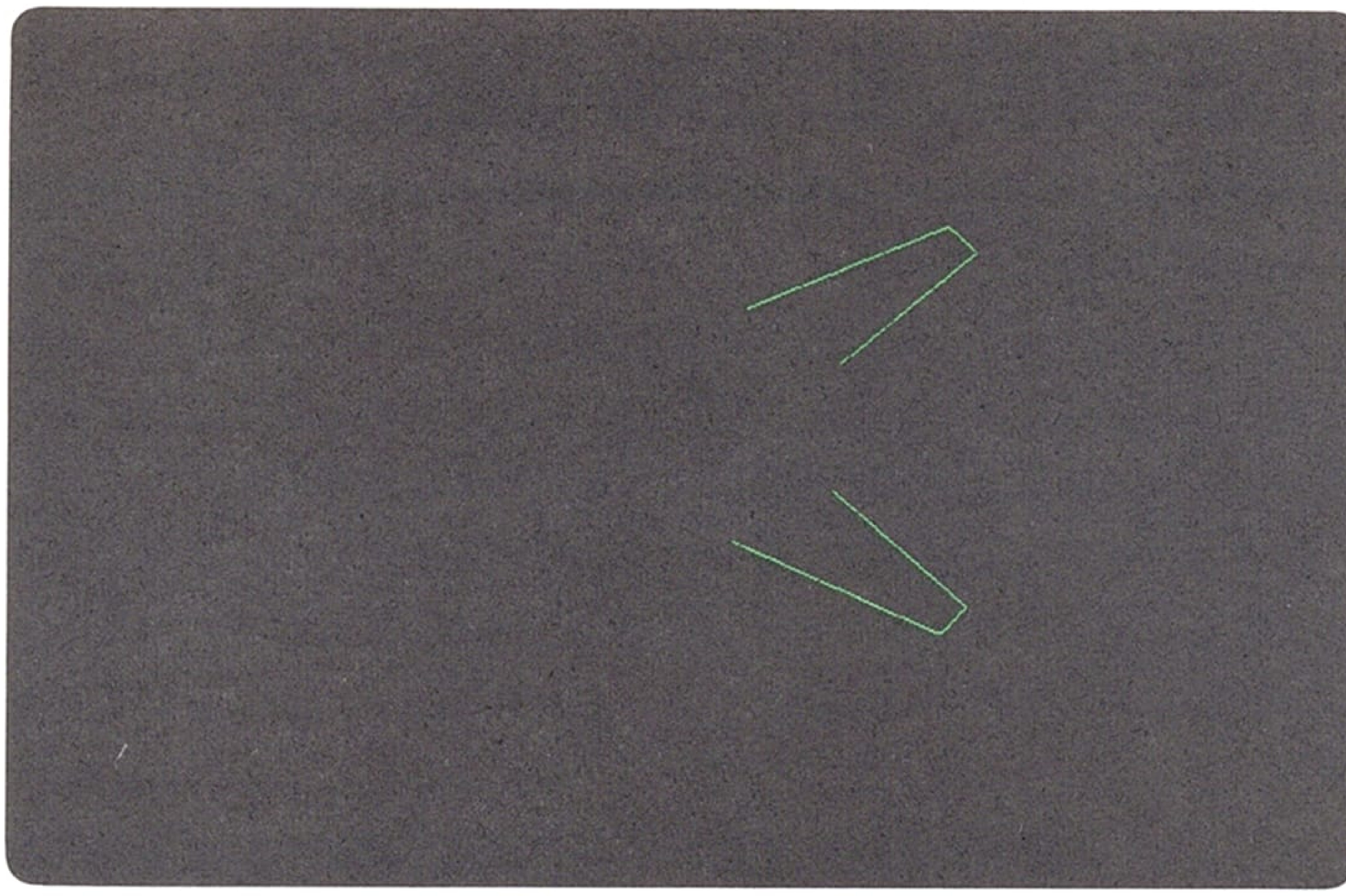
SCREEN 1,,,1



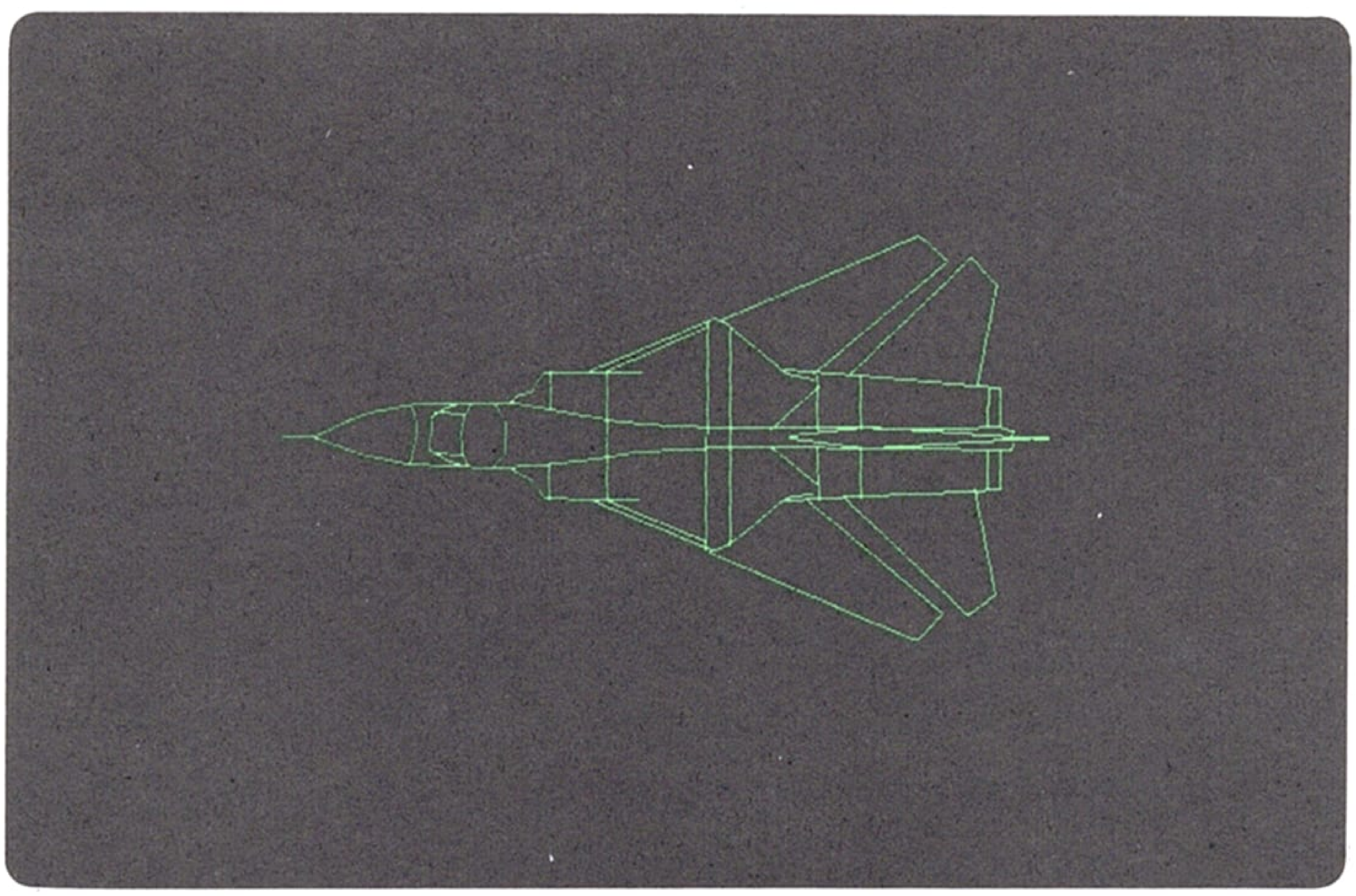
SCREEN 1,,,2



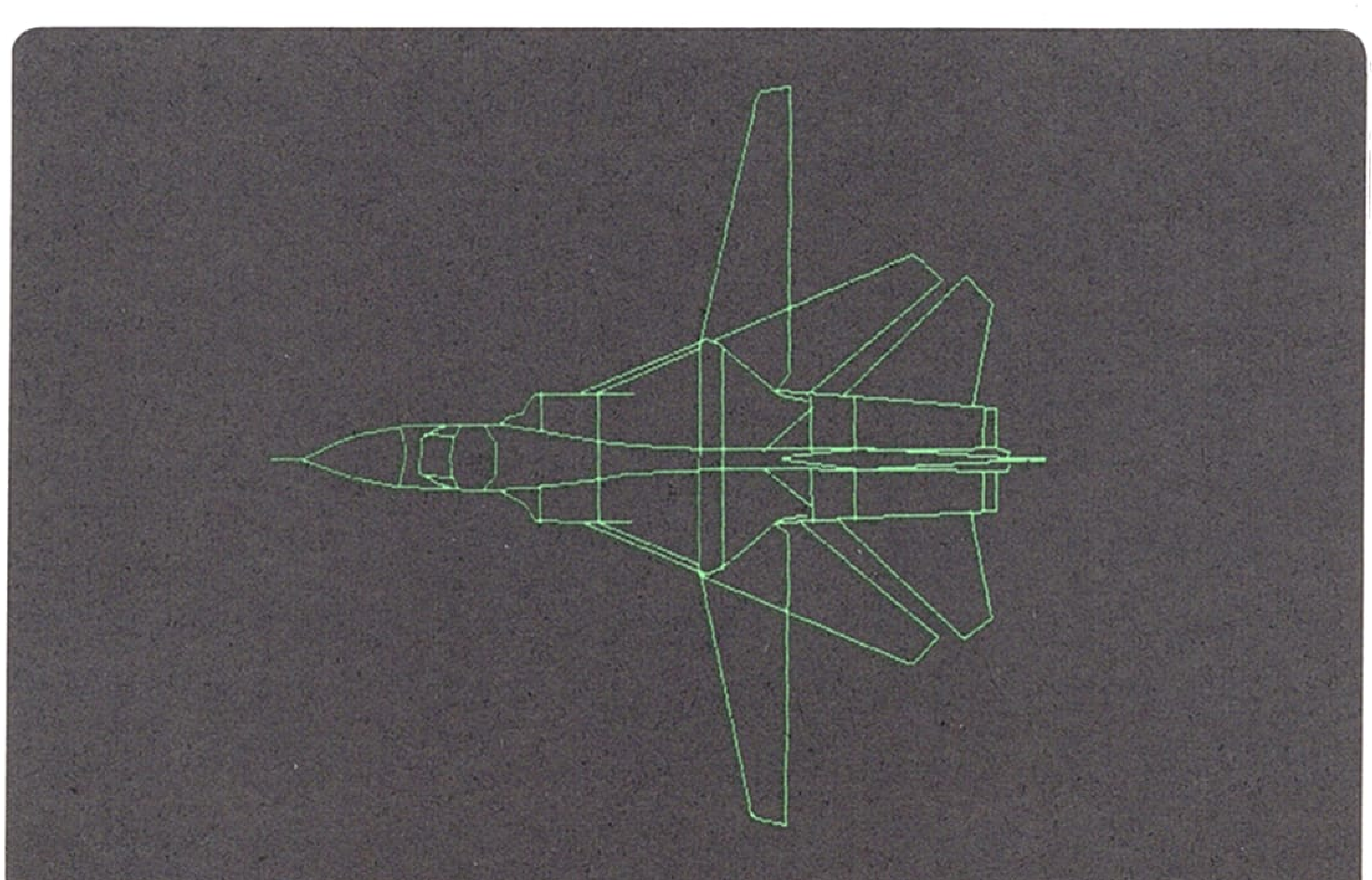
SCREEN 1,,,3



SCREEN 1,,,4



SCREEN 1,,,5



SCREEN 1,,,7

2·5 VIEW文を活用しよう

単純なLINE文だけでも，おもしろい絵がかけますし，労力さえいとわなければ，かなり迫真力のあるグラフィックスができることがおわかりいただけたでしょう。SCREENを使えばある程度の動きすら出せました。まことにPC-8801は頼りがいのある機械です。

私のパソコンおよびグラフィックスに対する考え方は，グラフィックスを単なるグラフィックスとして閉じた形にしておくことでなく，総合的なパソコン技術の一環としてとり入れたいということにあります。

たとえば，三面図からデータをとり込んでCRT画面上に図形を表示するだけではなくて，その3次元図形を実際に作りあげて行くところまでつきあいたいと考えています。これはもちろん夢のような所もありますが，絵に描いた餅は食えないので，絵に描いた餅を実際に食える所までパソコンにやらせたいと思っているのです。とは言え私は非力ですし，今すぐそういうシステムができるとは思っていませんが，何年か後には必ずそういうものを作り出してみたいと思っているのです。現に産業界ではそういうシステムが活躍しています。

何とかグラフィックスを物を作る方向へ引っ張り出せないかと考えていますと，こういう事に思い当りました。子供の頃，ボール紙を使ってしきりに工作をしたことがあります。図面や写真をみながら作るのですが，スケールの拡大がわずらわしく，最初のうちに精密に引いていた拡大図面も段々作る途中で粗雑になり，最後は写真とニラメッコをして，丸みをつけたり，ハサミでチョン切ったりしました。うまく丸みのでない所は和紙を何回も洗濯ノリで重ねバリして形を作りました。かなり熱中して腕が上がりましたが，図面に忠実なものはなかなか出ませんでした。その内にプラモデルという画期的な発明が出てきて，自分で図面をいじることはなくなりました。

この本を書きながら，突然20年以上も昔の経験を思い出して，本屋さんへ出かけて模型作りの棚を探してみました。何かヒントがないかなと思ったのです。すると非常に面白い本を見つけました。

　摺本好作著『切り抜く本』（誠文堂新光社）

という本です。これはシリーズ物になっていて、何冊か出ていましたが､この著者の本が一番私の目的にはあっていたようです｡それは対象が直線的に構成されているので作りやすいということと，出来上りは，この本のものが一番すぐれているように思えたからです。大胆に省略がなされていて，部品数が少ないことも歓迎した理由の一つです。

現物はボール紙に作ってあって，ノリシロがついているという面白いものです。１つを選んでディジタイザーで入力してみました。紙は工作が容易ですが，すぐこわれてしまうという欠点があるので，プロッタを使って銅板にケガキをさせると良いと思いました。

遊んでばかりいるとおしかりを受けそうですが，なかなかうまく入力できたと思います。出力にプリンタでなくプロッタを使えばもう少し迫真性がでただろうと思います。

これらの絵はワールド座標を使って描き，WINDOW 文を適当に使って読み出しています。読み出した図面を見ている内に，いくつか役に立つと思われるヒントを見つけましたので，記録しておきたいと思います。最初の図面には座標がありません。従って何が図面のどの辺に位置しているか，大変わかりにくいのです。

そこで，部品図は第１画面に描き，座標読みとり用の格子を第２画面に描くと便利だと思います。つまり図を描く時に次のようにSCREEN文を使いわけるのです。

SCREEN　1，，0，1 →部品図を書く

SCREEN　1，，1，2 →座標読みとり用の格子を書く

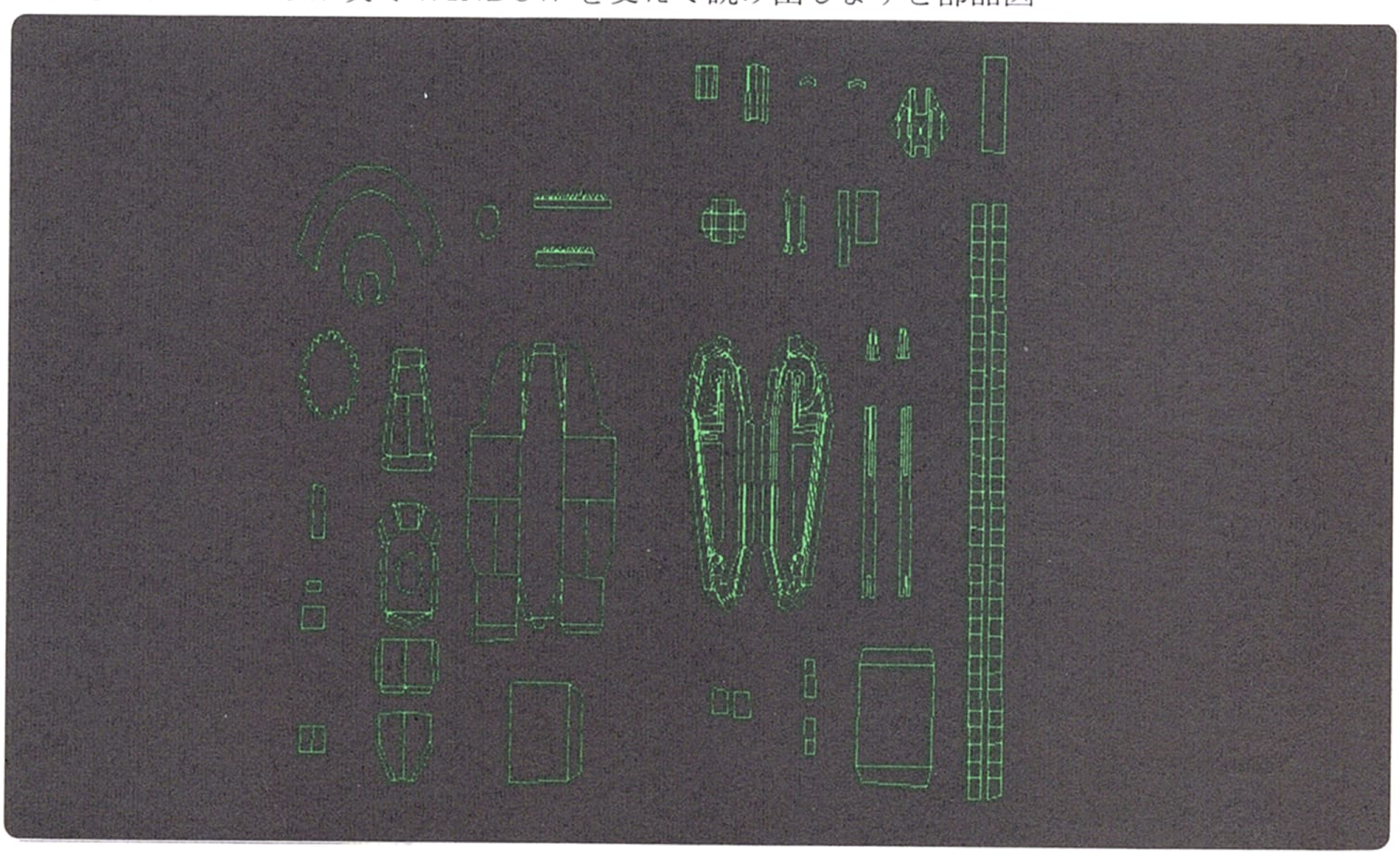
図を読み出す時にはSCREEN文を次のように使いわけます。

SCREEN　1，，，1 →部品図だけを見る

SCREEN　1，，，7 →座標の入った図面を見る

こうすると，どこにWINDOW をかけようかと迷った時に便利です。格子を色わけしておくともっと便利です。

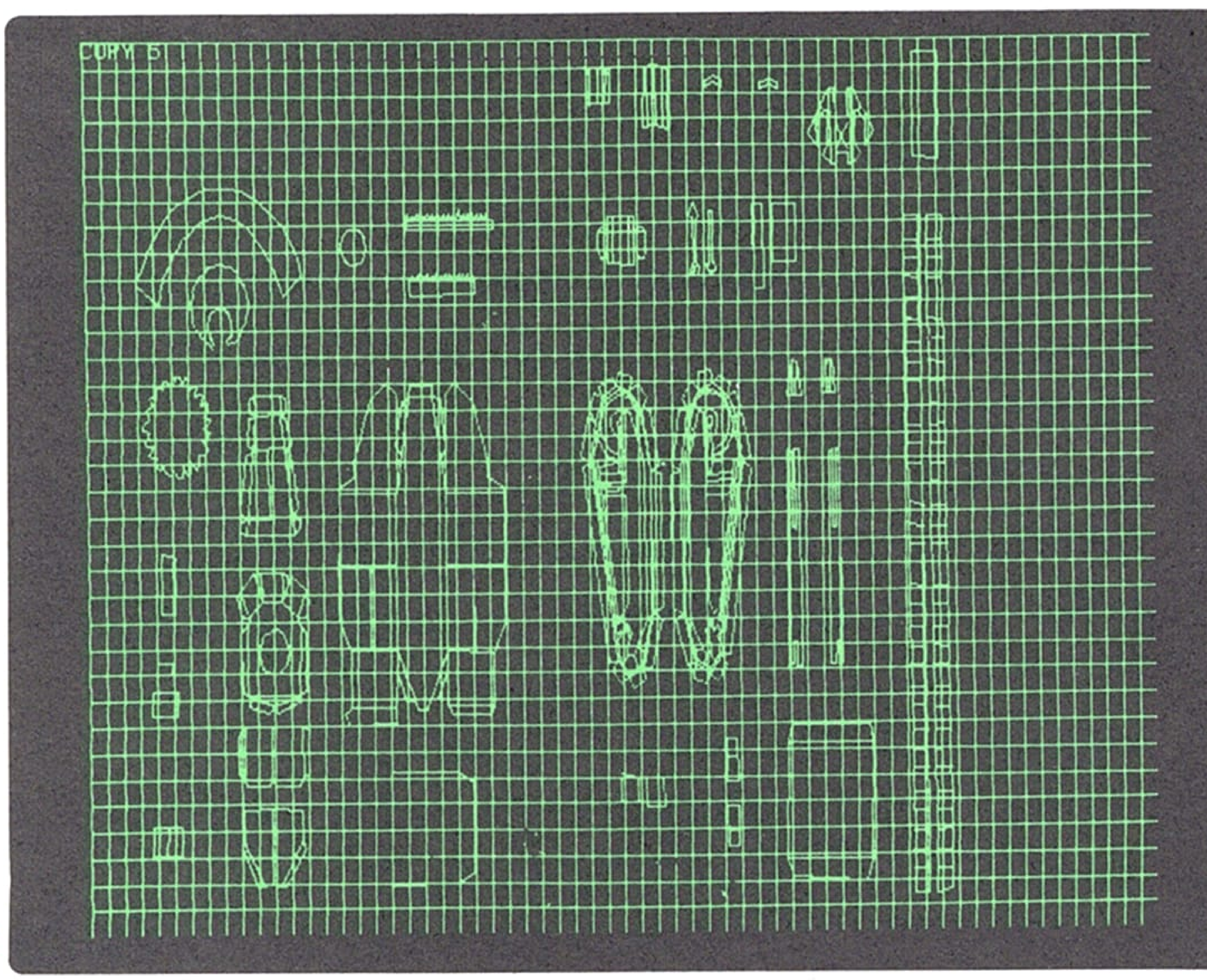
たとえばWINDOW 文でWINDOW を変えて読み出しますと部品図



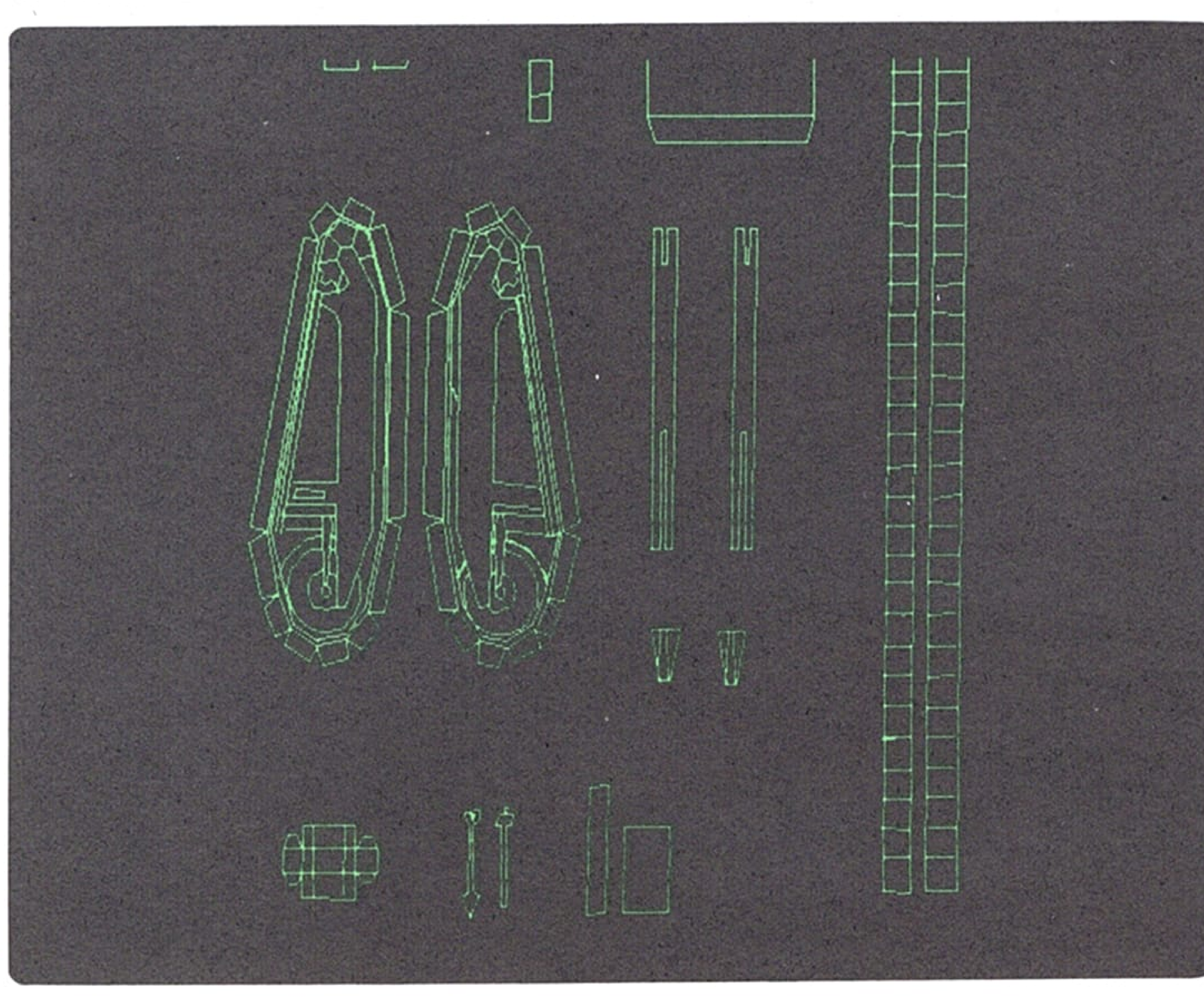
の一部を拡大することができます。これは大変都合の良いことです。実行した結果を下図に示しておきます。

しかし，それならばVIEW文を使って画面の一部に全部品図を示しておき，残りの画面に部品の拡大図を出してみたらどうだろうかと思いました。実際にやった結果を右ページに示します。なかなか面白いと思いました。VIEW文は使い方によっては便利です。

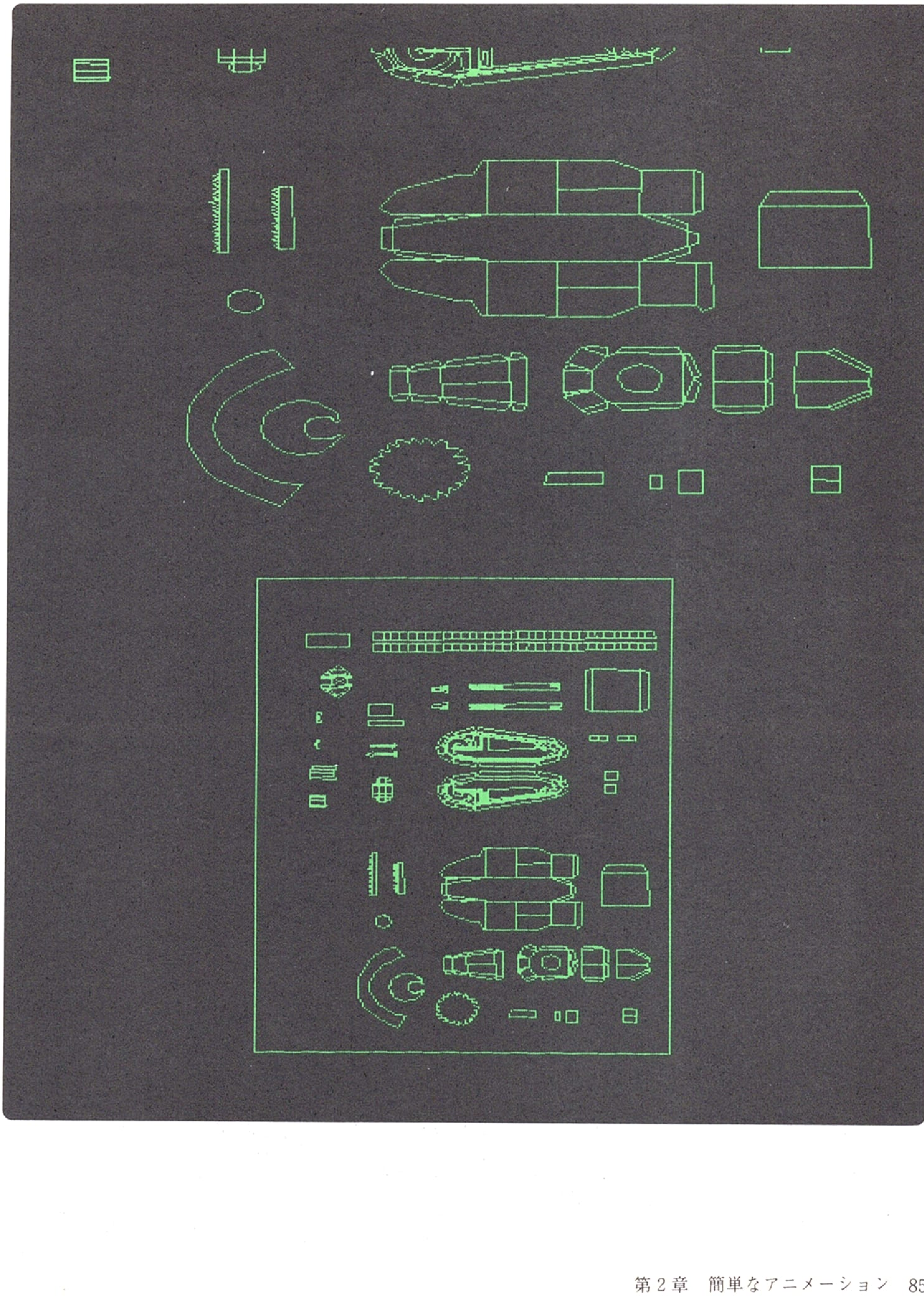
日本地図の全体を示しておき，その一部の関東地方を示し，東京都を示し千代田区を示し，私の大学を示すというようなこともできます。



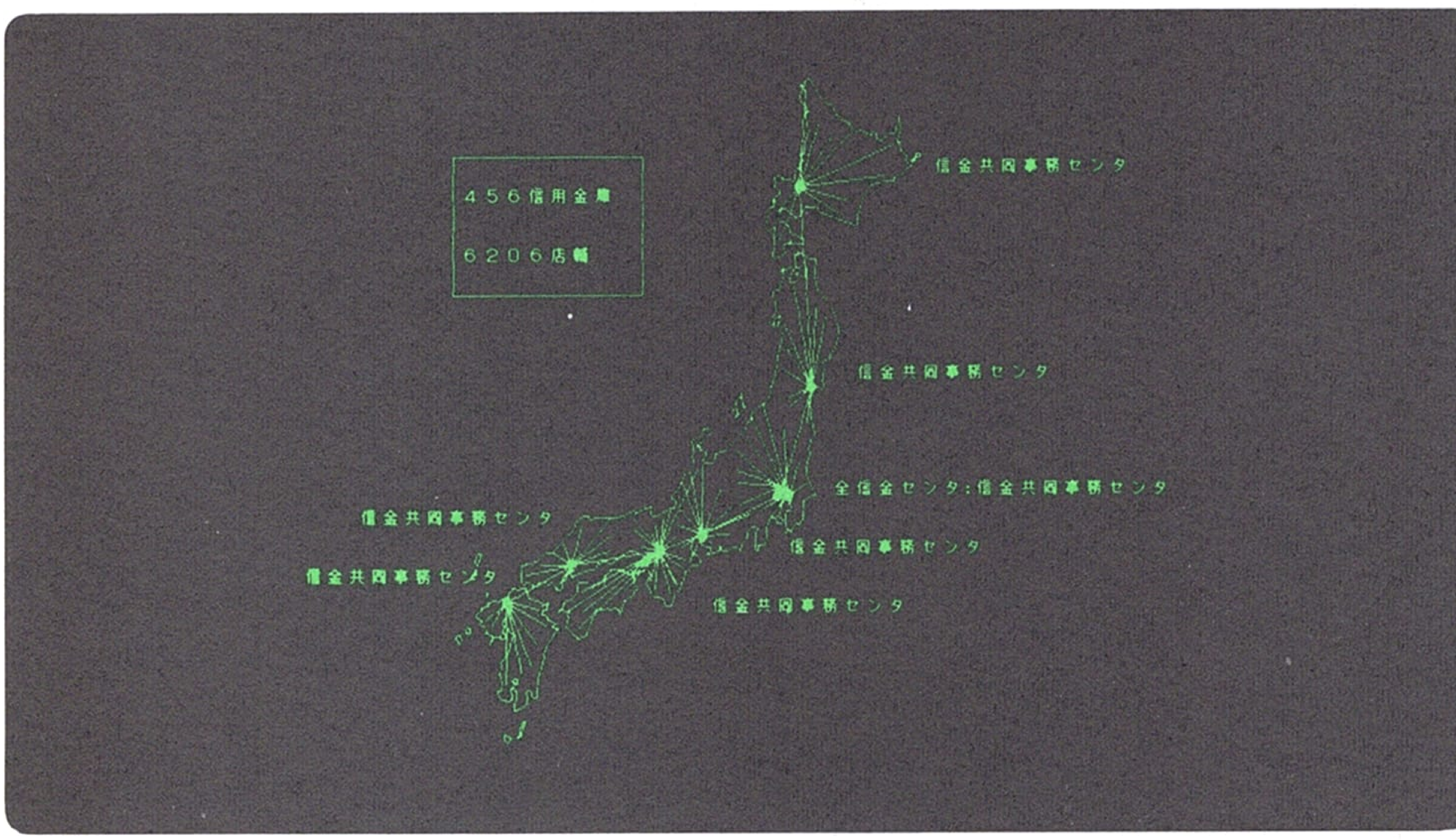
第２画面に格子を入れる→

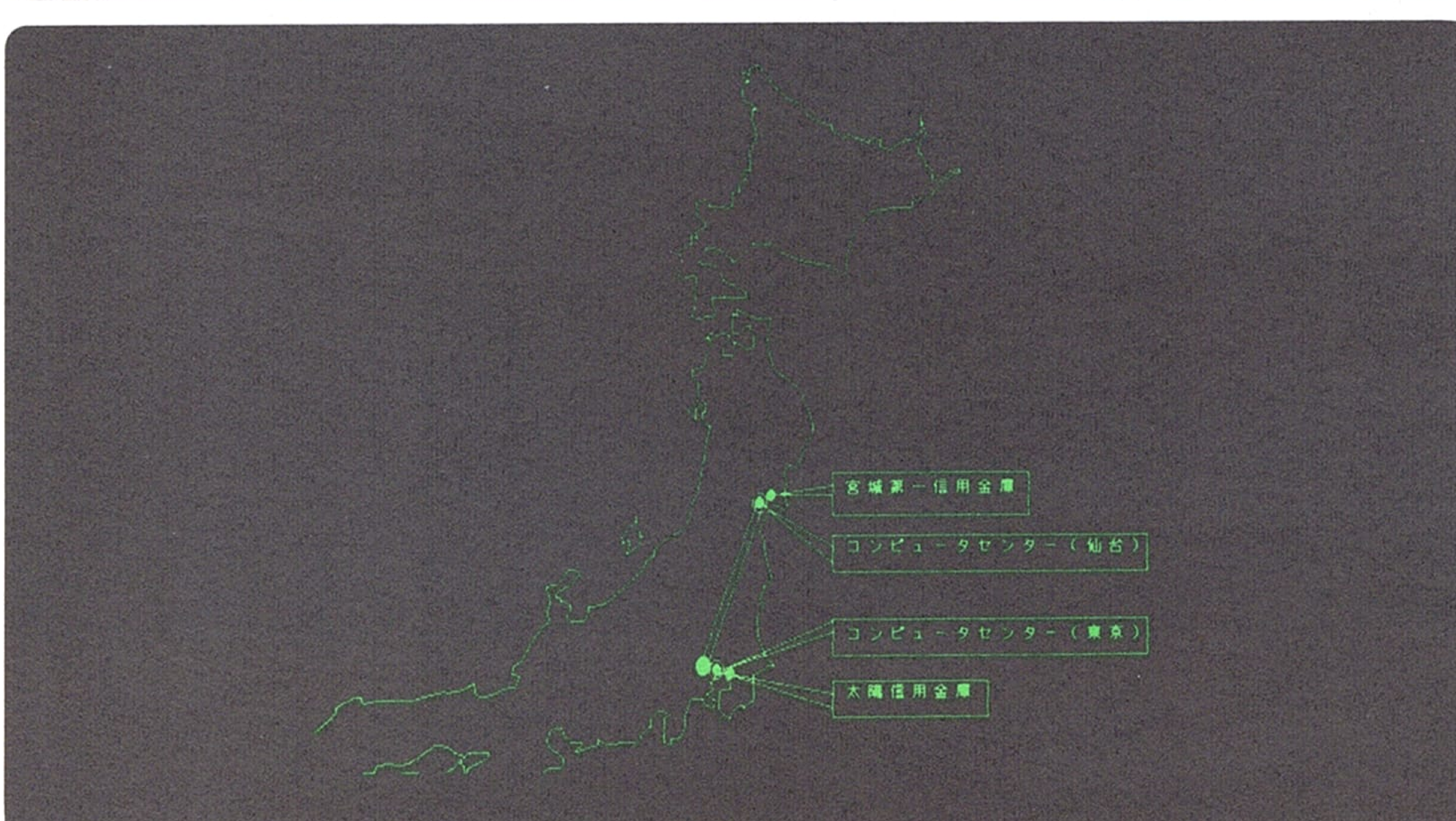


WINDOWを変えて一部を拡大→



VIEW文の応用として日本地図の全体と一部を利用した例をお見せしましょう。これはNHKテレビのために作ったものです。ハードコピーをとるために原図を簡略化してあります。色はとてもきれいに出たと思うのですが、NTSC式とRGB回路の変換器を用意していなかったために同期がとれず、テレビの画面用には面領域に色を塗ることは断念せざるを得ませんでした。画面での漢字のドット幅と、プリンタの漢字のドット幅が違うためにハードコピーは少し見にくくなっています。PC-８８０１は漢字が楽に扱えて大変楽だということを発見しました。





2·6　特別な入力機器を使わずに絵を描く

さて、本節ではディジタイザーなどの機械には一切頼らずにグラフィックスをする方法について考えてみましょう。全部人力に頼るのは大変つらいものですが、仕組を勉強するのにはもってこいです。

まず、どのような題材を扱うかについて決定します。決定したら資料を集めなくてはなりません。図面や写真をたくさん集めるのです。気に入った図面や写真がみつかったら、それを材料にします。しかしたいていの図面や写真はそのままでは使えないことが多いのです。図面は線が多すぎて、かえって困ることが多く、写真では大切な部分がカゲに入ってしまうことがあります。そこで必ず材料には手を加え、図面を引き直します。女の人だってアイラインをちょっと引いたり、口紅を少し使うと、おやっと見違えるくらいになるでしょう。そのままで悪いことはないのですが、ワンポイント手を加えると、グッと引き立ちます。

入力用の図面は方眼紙に書くか、透明な方眼紙で写して下さい。一度原図のコピーをとってから、ガラスの上におき下から光で照らして方眼紙に写すなどの工夫をすると良いと思います。

図面ができたら座標を記入します。いろいろな流儀が考えられますが、PC-８８０１の場合、WORLD座標系が使え、POINT文が使えることを考慮に入れますと、左側で一番先頭、しかも上下の中央部あたりを(0,0)に選んでおきますと、後で楽です。私の作った例を次ページにお目にかけましょう。この図は例題用なので写真からは作らず、私が勝手に創作しました。図からもわかる通り、できるだけ点の数をへらしてあります。

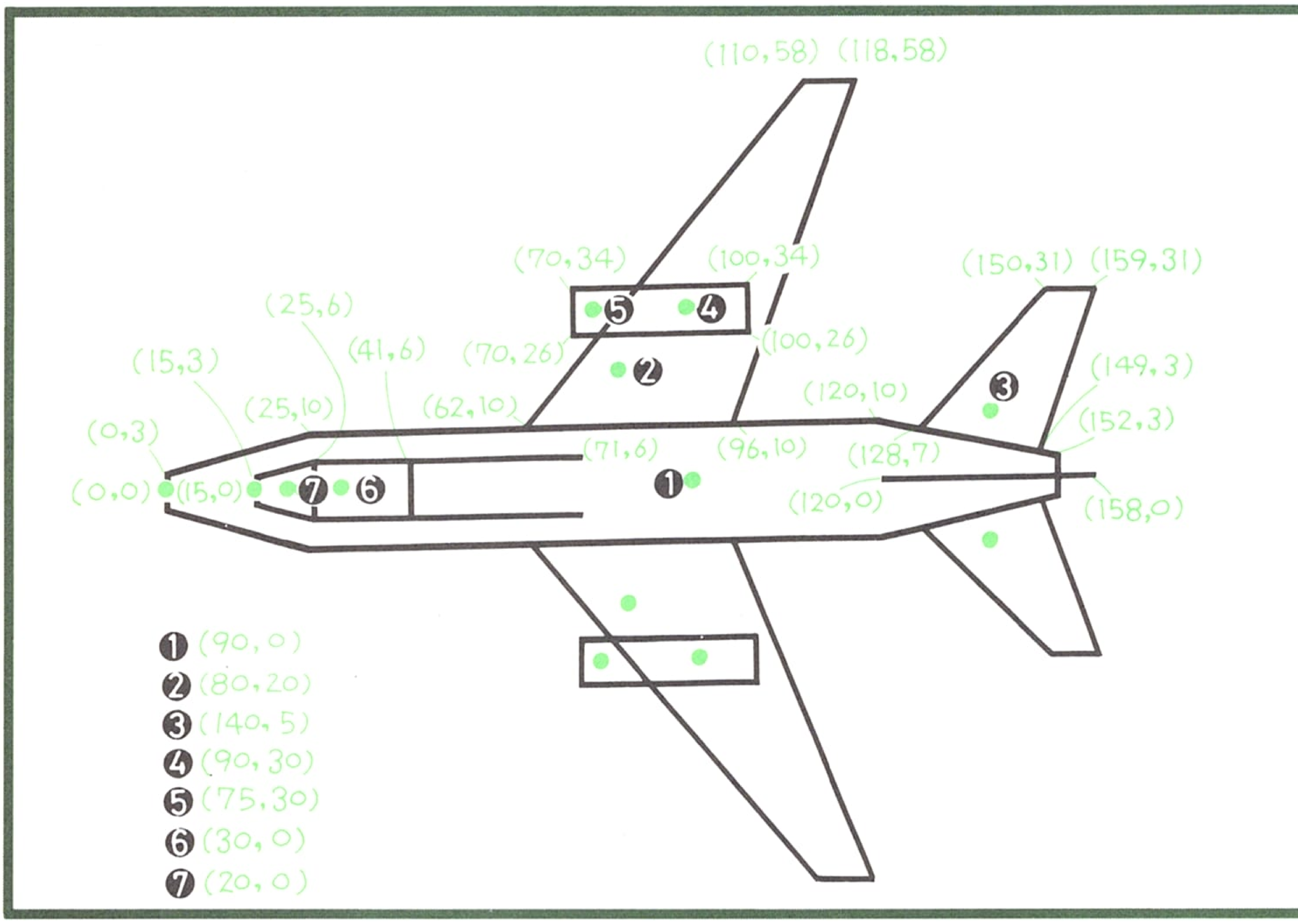
線を引くための座標だけでなく、色を塗るための座標も記録しておいて下さい。これを忘れてしまうと、後で色を塗る時、画面上で苦戦することになります。(原図というのは、しばしばなくなってしまうものなのです。)

座標をとり終ったら、線を描き始めます。丸い部分は円弧を使いたくなる誘惑にかられるでしょうが、なるべくさけて線分で近似して下さい。円はVIEW文やWINDOW文によって、同等の比率で縮小されず、失敗のもとになりやすいのです。

もっとも単純な考え方のプログラムを最初に示します(プログラム1)。これは非常に素直なプログラムで、悪い部分はないですし、１つのLINE文さえ作ってしまえば、行番号を変更して行くだけで、入力は楽なのですが、不満がないわけではありません。それはやはり手間がか

かることと、同じ点の座標を２回入力することと、このプログラムが非常に個別的な考え方で作られているので、応用しょうとする時に問題が生じてくることです。とは言え最初私は、このプログラムは悪い例だというつもりで作ったのですが、でぎてしまうとそう悪くもないのではないかと思っています。

プログラムの実行結果をプリンタでハードコピーに落としたものを示してあります。大体思うようなものができています。

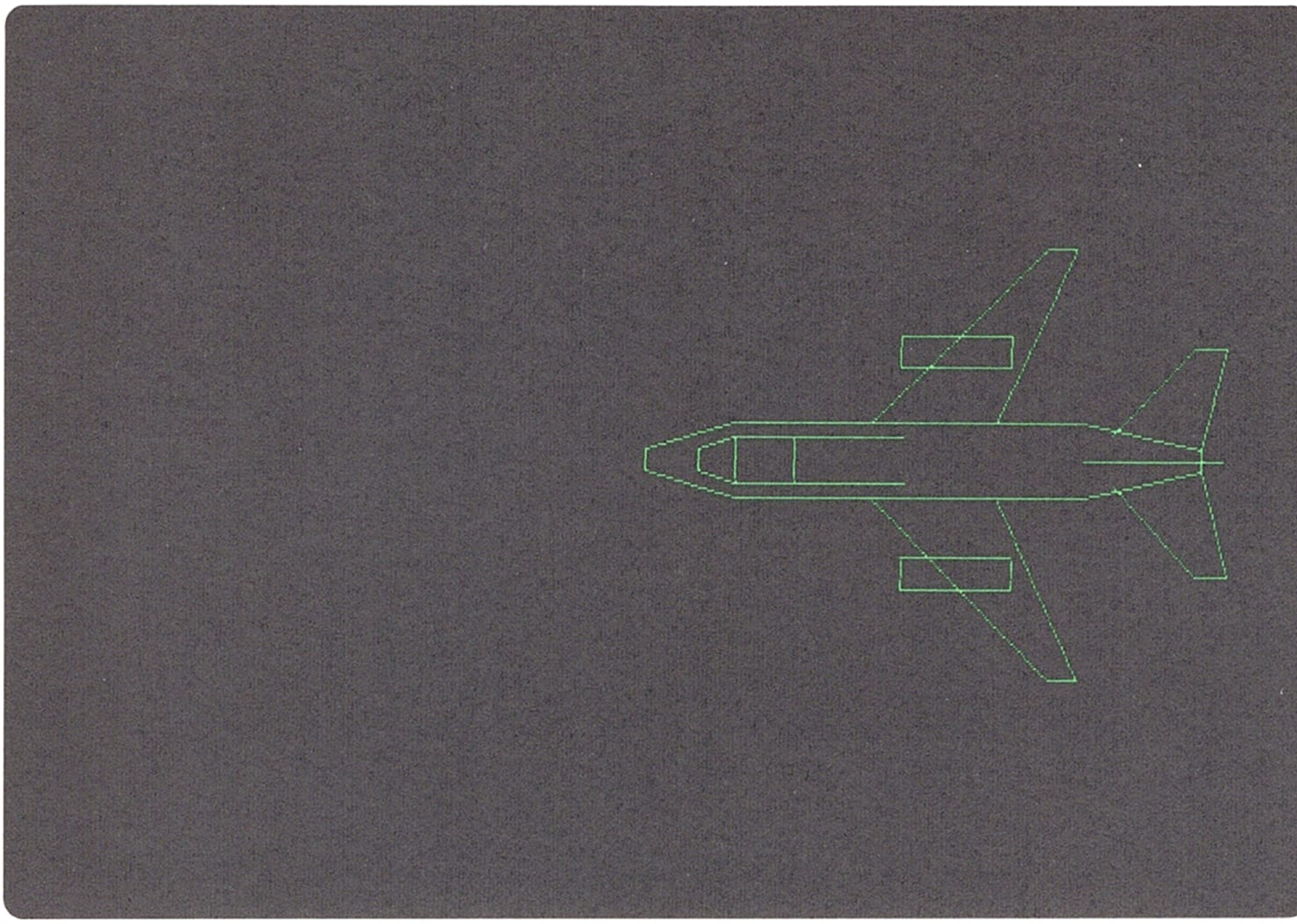


●プログラム１●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 WINDOW (-160,-100)-(160,100)
130 VIEW (0,0)-(639,199)
140 LINE (   0,   0)-(   0,   3),7
150 LINE (   0,   3)-(  25,  10),7
160 LINE (  25,  10)-(  62,  10),7
170 LINE (  62,  10)-( 120,  10),7
180 LINE ( 120,  10)-( 152,   3),7
190 LINE ( 152,   3)-( 152,   0),7
200 LINE (   0,   0)-(   0,  -3),7
210 LINE (   0,  -3)-(  25, -10),7
```

胴体部分を描く（140～210行）

```
220 LINE (  25, -10)-(  62, -10),7
230 LINE (  62, -10)-( 120, -10),7
240 LINE ( 120, -10)-( 152,  -3),7
250 LINE ( 152,  -3)-( 152,   0),7
260 '
270 LINE (  62,  10)-( 110,  58),7
280 LINE ( 110,  58)-( 118,  58),7
290 LINE ( 118,  58)-(  96,  10),7
300 LINE ( 128,   7)-( 150,  31),7
310 LINE ( 150,  31)-( 159,  31),7
320 LINE ( 159,  31)-( 152,   3),7      主翼部分を描く
330 LINE (  62, -10)-( 110, -58),7
340 LINE ( 110, -58)-( 118, -58),7
350 LINE ( 118, -58)-(  96, -10),7
360 LINE ( 128,  -7)-( 150, -31),7
370 LINE ( 150, -31)-( 159, -31),7
380 LINE ( 159, -31)-( 152,  -3),7
390 '
400 LINE ( 120,   0)-( 158,   0),7      垂直尾翼部分を描く
410 '
420 LINE (  15,   0)-(  15,   3),7
430 LINE (  15,   3)-(  25,   6),7
440 LINE (  25,   6)-(  71,   6),7      水平尾翼部分を描く
450 LINE (  15,   0)-(  15,  -3),7
460 LINE (  15,  -3)-(  25,  -6),7
470 LINE (  25,  -6)-(  71,  -6),7
480 LINE (  25,   6)-(  25,  -6),7      風防部分を描く
490 LINE (  41,   6)-(  41,  -6),7
500 '
510 LINE (  70,  34)-( 100,  26),7,B    燃料タンク部分を描く
520 LINE (  70, -34)-( 100, -26),7,B
530 '
540 END
```



同じ座標を２回出す煩わしさを省いたプログラムを次に作ってみました。作ってみますと前のプログラムよりシステマティックでないので、かえって面倒なところもありますが、どこから書き始めているかが良くわかるようになりました。

●プログラム２●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 WINDOW (-160,-100)-(160,100)
130 VIEW (0,0)-(639,199)
140 LINE (   0,   0)-(   0,   3),7
150 LINE-(  25,  10),7
160 LINE-(  62,  10),7
170 LINE-( 120,  10),7
180 LINE-( 152,   3),7
190 LINE-( 152,   0),7
200 LINE (   0,   0)-(   0,  -3),7
210 LINE-(  25, -10),7
220 LINE-(  62, -10),7
230 LINE-( 120, -10),7
240 LINE-( 152,  -3),7
```

```
250 LINE-( 152,    0),7
260 '
270 LINE (  62,   10)-( 110,   58),7
280 LINE-( 118,   58),7
290 LINE-(  96,   10),7
300 LINE ( 128,    7)-( 150,   31),7
310 LINE-( 159,   31),7
320 LINE-( 152,    3),7
330 LINE (  62,  -10)-( 110,  -58),7
340 LINE-( 118,  -58),7
350 LINE-(  96,  -10),7
360 LINE ( 128,   -7)-( 150,  -31),7
370 LINE-( 159,  -31),7
380 LINE-( 152,   -3),7
390 '
400 LINE ( 120,    0)-( 158,    0),7
410 '
420 LINE (  15,    0)-(  15,    3),7
430 LINE-(  25,    6),7
440 LINE-(  71,    6),7
450 LINE (  15,    0)-(  15,   -3),7
460 LINE-(  25,   -6),7
470 LINE-(  71,   -6),7
480 LINE (  25,    6)-(  25,   -6),7
490 LINE (  41,    6)-(  41,   -6),7
500 '
510 LINE (  70,   34)-( 100,   26),7,B
520 LINE (  70,  -34)-( 100,  -26),7,B
530 '
540 END
```

私の個人的な趣味としては、プログラム1、プログラム2よりも、DATA文を使うプログラムの方が好きです（**プログラム3**）。それはメインプログラムが非常に短くてすみ、図形の変更や、応用が大変自由にできるからです。

DATA文の作り方は次のようにしてあります。

　DATA　(X座標), (Y座標), (フラグ)

フラグは次のように使うように定めました。

　フラグが1の時――書き始めの点
　フラグが0の時――途中の点
　フラグが4の時――データの終りを示す

書き始めの点はPSETで処理し，途中の線分をLINE文で描いています。

●プログラム3●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 WINDOW (-160,-100)-(160,100)
130 VIEW (0,0)-(639,199)
140 COL=7
150 READ X,Y,F ←データを読み込む
160 IF F=4 THEN END ←フラグが4なら終り
170 IF F=1 THEN PSET(X,Y),COL:GOTO 190
180 LINE(XOLD,YOLD)-(X,Y),COL ←線を描く
190 XOLD=X:YOLD=Y ←次に線を描くための始点を変数に代入
200 GOTO 150
210 '
220 DATA      0,     0,     1
230 DATA      0,     3,     0
240 DATA     25,    10,     0
250 DATA     62,    10,     0
260 DATA    120,    10,     0
270 DATA    152,     3,     0
280 DATA    152,     0,     0
290 DATA      0,     0,     1
300 DATA      0,    -3,     0
310 DATA     25,   -10,     0
320 DATA     62,   -10,     0
330 DATA    120,   -10,     0
340 DATA    152,    -3,     0
350 DATA    152,     0,     0
360 DATA     62,    10,     1
370 DATA    110,    58,     0
380 DATA    118,    58,     0
390 DATA     96,    10,     0
400 DATA    128,     7,     1
410 DATA    150,    31,     0
420 DATA    159,    31,     0
430 DATA    152,     3,     0
440 DATA     62,   -10,     1
450 DATA    110,   -58,     0
460 DATA    118,   -58,     0
470 DATA     96,   -10,     0
480 DATA    128,    -7,     1
490 DATA    150,   -31,     0
500 DATA    159,   -31,     0
510 DATA    152,    -3,     0
```

```
520 DATA        120,        0,        1
530 DATA        158,        0,        0
540 DATA         15,        0,        1
550 DATA         15,        3,        0
560 DATA         25,        6,        0
570 DATA         71,        6,        0
580 DATA         15,        0,        1
590 DATA         15,       -3,        0
600 DATA         25,       -6,        0
610 DATA         71,       -6,        0
620 DATA         25,        6,        1
630 DATA         25,       -6,        0
640 DATA         41,        6,        1
650 DATA         41,       -6,        0
660 DATA         70,       34,        1
670 DATA        100,       34,        0
680 DATA        100,       26,        0
690 DATA         70,       26,        0
700 DATA         70,       34,        0
710 DATA         70,      -34,        1
720 DATA        100,      -34,        0
730 DATA        100,      -26,        0
740 DATA         70,      -26,        0
750 DATA         70,      -34,        0
760 DATA        999,      999,        4
```

この方法がどんなに便利かを示すのが次のプログラム4です。わずかな変更だけで一挙に4機編隊を実現することができます。先ほどのLINE文のプログラムですと、できないわけではありませんが、相当大幅な手直しが必要となる上に、長くなってしまいます。DATA文はプログラム3とまったく同じで、一切手を加えていません。つまり、グラフィックスにおいて重要な役割を持っているのはメインプログラムの部分でDATA文は副次的な役割しか果たしていないことがおわかりいただけると思います。

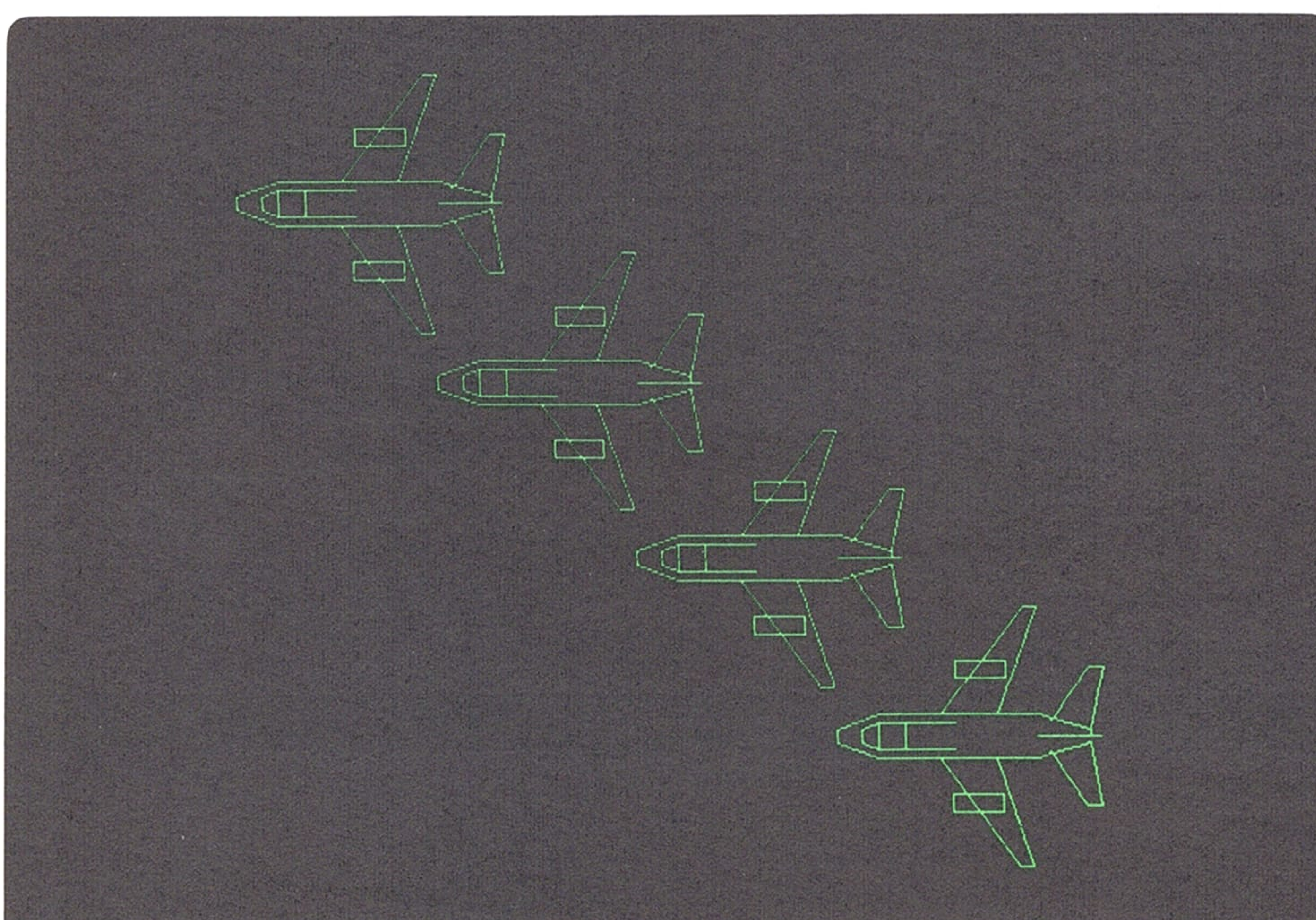
●プログラム4●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE  0,25,0,1
120 WINDOW (0,0)-(640,400)  ←ウィンドウを変更
```

```
130 VIEW (0,0)-(639,199)
140 COL=7
150 FOR I=1 TO 4
160 X0=120*I:Y0=80*I ←ループ1回につきヨコ軸タテ軸方向に
                        120,80ドットずらす
170 READ X,Y,F
180 IF F=4 THEN GOTO 230
190 IF F=1 THEN PSET (X+X0,Y+Y0),COL:GOTO 210
200 LINE(XOLD+X0,YOLD+Y0)-(X+X0,Y+Y0),COL
210 XOLD=X:YOLD=Y
220 GOTO 170
230 RESTORE ←読み込むデータを先頭にもどして次のループへ
240 NEXT I
250 '
260 DATA        0,      0,      1
270 DATA        0,      3,      0
280 DATA       25,     10,      0
```

●290行以降のDATA文はプログラム3の250行以降と同じなので省略してあります。

プログラム4の実行結果を次に示します。



もうひとつ現実の世界ではありえないことをやってみましょう。

メインプログラムを次のプログラムのように書きかえて見て下さい。COLORパレットの変更によって飛行機が斜めに前進したり後退したりします。DATA文は行番号をずらすだけで良く、手を加える必要はありません。

●プログラム5●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 WINDOW (0,0)-(640,400)
130 VIEW (0,0)-(639,199)
140 COL=7
150 FOR I=1 TO 4
160 X0=120*I:Y0=80*I
170 READ X,Y,F
180 IF F=4 THEN GOTO 230
190 IF F=1 THEN PSET (X+X0,Y+Y0),I   :GOTO 210
200 LINE(XOLD+X0,YOLD+Y0)-(X+X0,Y+Y0),I
210 XOLD=X:YOLD=Y
220 GOTO 170
230 RESTORE
240 NEXT I
250 '
260 COLOR=(1,7):COLOR=(2,0):COLOR=(3,0):COLOR=(4,0)
270 FOR I=1 TO 200 :NEXT I
280 COLOR=(1,0):COLOR=(2,7):COLOR=(3,0):COLOR=(4,0)
290 FOR I=1 TO 200 :NEXT I
300 COLOR=(1,0):COLOR=(2,0):COLOR=(3,7):COLOR=(4,0)
310 FOR I=1 TO 200 :NEXT I
320 COLOR=(1,0):COLOR=(2,0):COLOR=(3,0):COLOR=(4,7)
330 FOR I=1 TO 200 :NEXT I
340 GOTO 260
350 '
360 DATA        0,       0,       1
370 DATA        0,       3,       0
380 DATA       25,      10,       0
390 DATA       62,      10,       0
400 DATA      120,      10,       0
```

●410行以降のDATA文はプログラム3の270行以降と同じです。

ついでに色を塗ってしまうところまでDATA文を使う方式でやってみたのが次のプログラムで。DATA文の作り方は次のようにしてあります。

DATA　(X座標), (Y座標), (カラー), (フラグ)

フラグの使い方は次のように定めました。

フラグが0の時――データの途中

フラグが4の時――データの終り

●プログラム6●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 WINDOW (-160,-100)-(160,100)
130 VIEW (0,0)-(639,199)
140 COL=7
150 READ X,Y,F
160 IF F=4 THEN GOTO 220
170 IF F=1 THEN PSET(X,Y),COL:GOTO 190
180 LINE(XOLD,YOLD)-(X,Y),COL
190 XOLD=X:YOLD=Y
200 GOTO 150
210 '
220 READ X,Y,C,F
230 IF F=4 THEN END
240 PAINT (X,Y),C,COL
250 GOTO 220
260 '
270 DATA       0,      0,      1
280 DATA       0,      3,      0
290 DATA      25,     10,      0
300 DATA      62,     10,      0
310 DATA     120,     10,      0
320 DATA     152,      3,      0
330 DATA     152,      0,      0
340 DATA       0,      0,      1
350 DATA       0,     -3,      0
360 DATA      25,    -10,      0
370 DATA      62,    -10,      0
380 DATA     120,    -10,      0
390 DATA     152,     -3,      0
400 DATA     152,      0,      0
410 DATA      62,     10,      1
```

```
420 DATA      110,     58,      0
430 DATA      118,     58,      0
440 DATA       96,     10,      0
450 DATA      128,      7,      1
460 DATA      150,     31,      0
470 DATA      159,     31,      0
480 DATA      152,      3,      0
490 DATA       62,    -10,      1
500 DATA      110,    -58,      0
510 DATA      118,    -58,      0
520 DATA       96,    -10,      0
530 DATA      128,     -7,      1
540 DATA      150,    -31,      0
550 DATA      159,    -31,      0
560 DATA      152,     -3,      0
570 DATA      120,      0,      1
580 DATA      158,      0,      0
590 DATA       15,      0,      1
600 DATA       15,      3,      0
610 DATA       25,      6,      0
620 DATA       71,      6,      0
630 DATA       15,      0,      1
640 DATA       15,     -3,      0
650 DATA       25,     -6,      0
660 DATA       71,     -6,      0
670 DATA       25,      6,      1
680 DATA       25,     -6,      0
690 DATA       41,      6,      1
700 DATA       41,     -6,      0
710 DATA       70,     34,      1
720 DATA      100,     34,      0
730 DATA      100,     26,      0
740 DATA       70,     26,      0
750 DATA       70,     34,      0
760 DATA       70,    -34,      1
770 DATA      100,    -34,      0
780 DATA      100,    -26,      0
790 DATA       70,    -26,      0
800 DATA       70,    -34,      0
810 DATA      999,    999,      4
820 '
830 DATA       90,      0,      5,       0
840 DATA       80,     20,      4,       0
```

```
850 DATA        140,     15,      4,      0
860 DATA         90,     30,      5,      0
870 DATA         75,     30,      2,      0
880 DATA         30,      0,      1,      0
890 DATA         20,      0,      1,      0
900 DATA         80,    -20,      4,      0
910 DATA        140,    -15,      4,      0
920 DATA         90,    -30,      5,      0
930 DATA         75,    -30,      2,      0
940 DATA        999,    999,    999,      4
```

これで第2章は終りです。これまで勉強したことで、2次元図形を描いたり、動かしたりするのに必要なテクニックは大体マスターできたと思います。ここまでのテクニックでも面白いことができます。しかし、色の使い方が7色だけですし、2次元図形に限られています。もっとたくさんの色が使いたいと思いますし、グラフィックスの中でも最も魅力的な3次元グラフィックスが残っています。次章ではまず色の使い方からマスターしましょう。

第3章

色の使い方



3

3·0 ラインスタイルとは何だろう

色の塗り方を勉強する前に、線の引き方についてもう少し勉強しておきましょう。面を塗ることは2次元的領域を塗ることですが、線を引くことは1次元的領域を塗ることだと言うこともできるでしょう。従って線の引き方を良く勉強しておくことは、色の塗り方の理解をも深めることになります。まずLINE文の構文図を見て下さい。

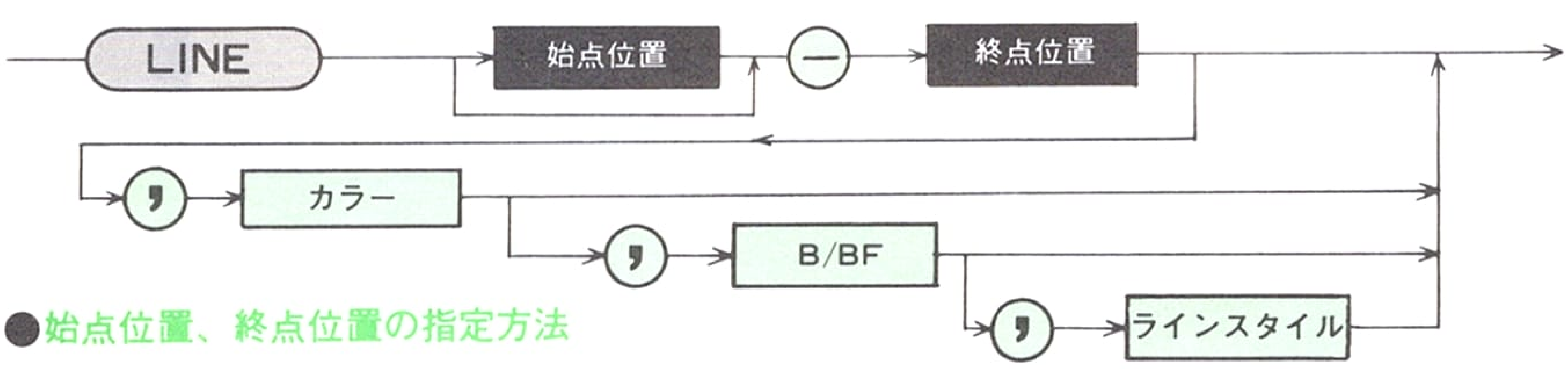
ラインスタイルというおかしなものが入っているでしょう。

LINE ●line／ライン

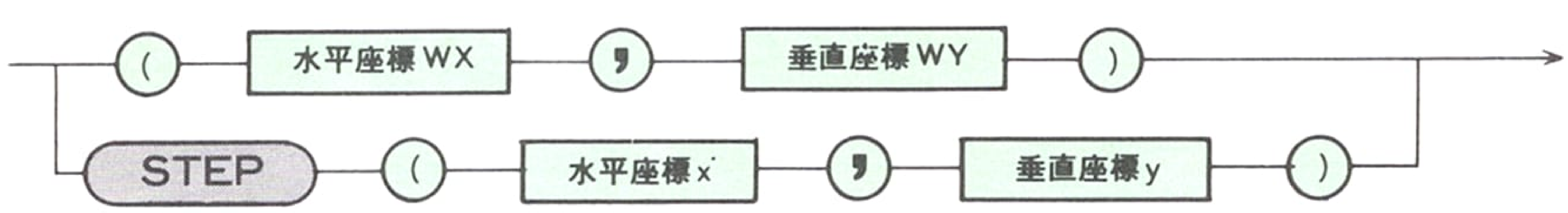
目的 線を描く

書式
LINE〔|　　(WX, WY)|〕－|　　(x, y)|
　　　|STEP (WX, WY)|　　|STEP (x, y)|
〔,〈パレット番号〉〕〔,|B |〕〔,〈ラインスタイル〉〕
　　　　　　　　　　　　|BF|

文例 LINE (100,100) － (135,100) ,7, ,&HF99F



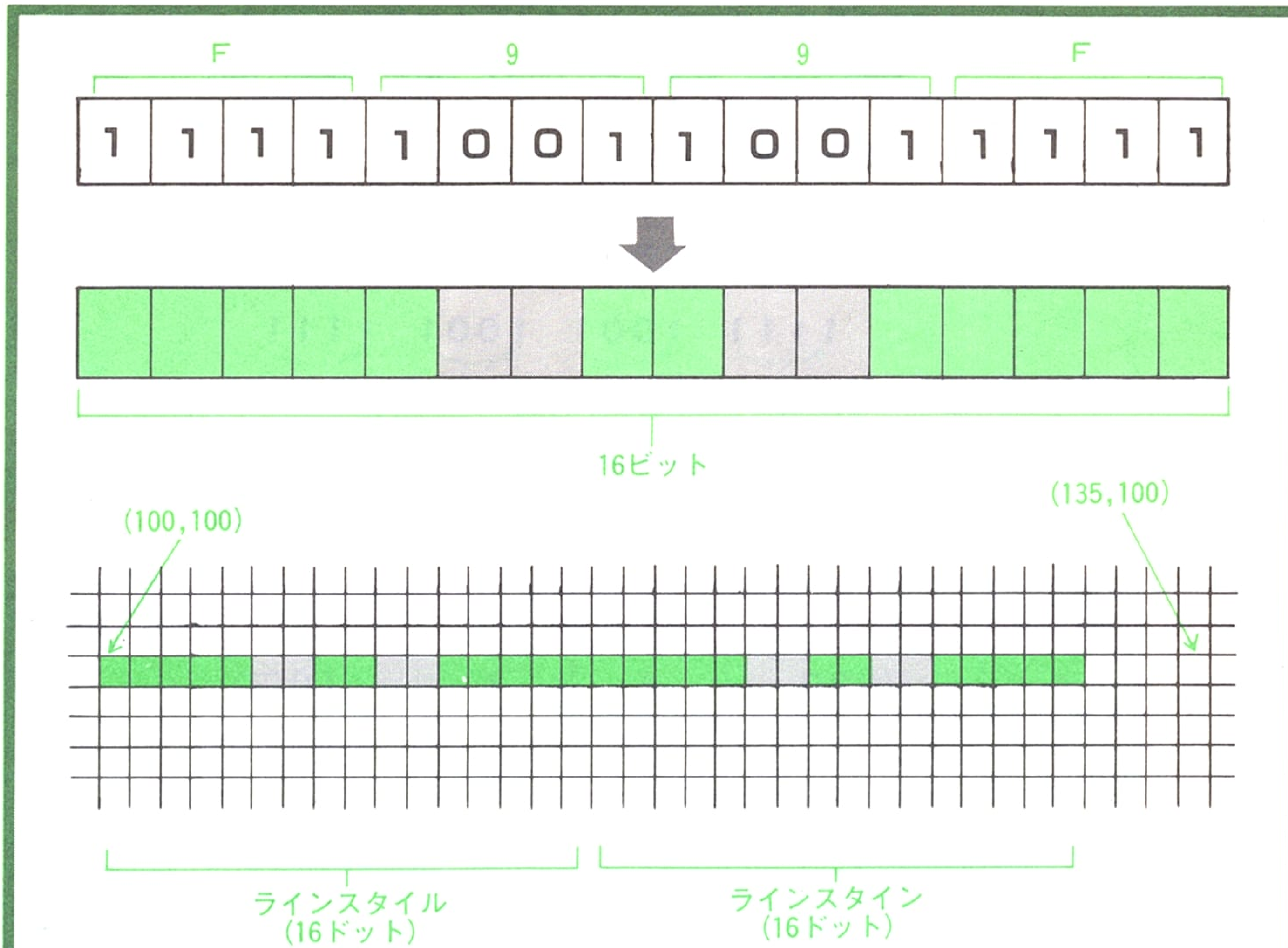
●始点位置、終点位置の指定方法



これまで線を引くのに使っていたのは、実線と言って切れ目のないものばかりでした。ところがラインスタイルを使いますと、実線ばかりでなく、破線や一点鎖線や点線まで 65537通りの線が表現できることになっています。どうすれば、そんなことが可能なのでしょうか。ここに、ラインスタイルというものが登場してきます。

ラインスタイルは16ビットまでの周期で任意のパターンを作ることを許します。16ビットは16ドットといいかえてもかまいません。次ページの図を見て下さい。16個のマス目があって、その中に1と0が書き込んでありますね。このマス目1つ1つがCRT画面上のドット1つ1つに対応し、同時にプリンタの出力用紙上のドット1つ1つに対応しています。

それでは1や0は何を意味しているのでしょうか。大ざっぱにいい



LINE (100,100) － (135,100) ,7, ,& HF99F を実行した時

ますと、1の時は点を表示し、0の時は点を表示しないことになるのです。これは図を見ればわかるでしょう。

次に問題となるのはラインスタイルの指定法です。16ビットをそのまま2進法で表現しますと、たとえば次のようになり、長すぎてかないません。

1111　1001　1001　1111

これが10進法ですと、もっと短くなります。しかし、この場合、直観的にパターンを思い浮かべるのが面倒になります。

ところが、16進法ですと使いやすい上に、さらに短くなります(&Hは16進数をあらわす符号です)。

&HF99F

そこでラインスタイルを指定する時には16進数を使うと良いことになります(10進数でもかまいません)。

●2進数の１１１１　１００１　１００１　１１１１が、16進数の&HF９９Fになることについて（進んだ読者は飛ばして下さい）

２進数を16進数に直すやり方がわからないという方もおられるでしょう。簡単に説明しておきましょう。

まず最下位の桁から数えて、４桁ずつ２進数を区切ります。

１１１１　１００１　１００１　１１１１
　４　　　４　　　４　　　４

これから次の対応表を使って４桁の２進数を16進数におきかえます。上の２進数が＆ＨＦ９９Ｆになることを確かめて下さい。

10進数	2進数	16進数	10進数	2進数	16進数
0	0000	0	8	1000	8
1	0001	1	9	1001	9
2	0010	2	10	1010	A
3	0011	3	11	1011	B
4	0100	4	12	1100	C
5	0101	5	13	1101	D
6	0110	6	14	1110	E
7	0111	7	15	1111	F

それではラインスタイルを使っていろいろな線を引いてみましょう。線を引く時にSTEP文を使うと、楽にプログラミングができます。線と線の間は10ドット位離すようにして下さい。そうでないと重なってわかりにくくなってしまいます。

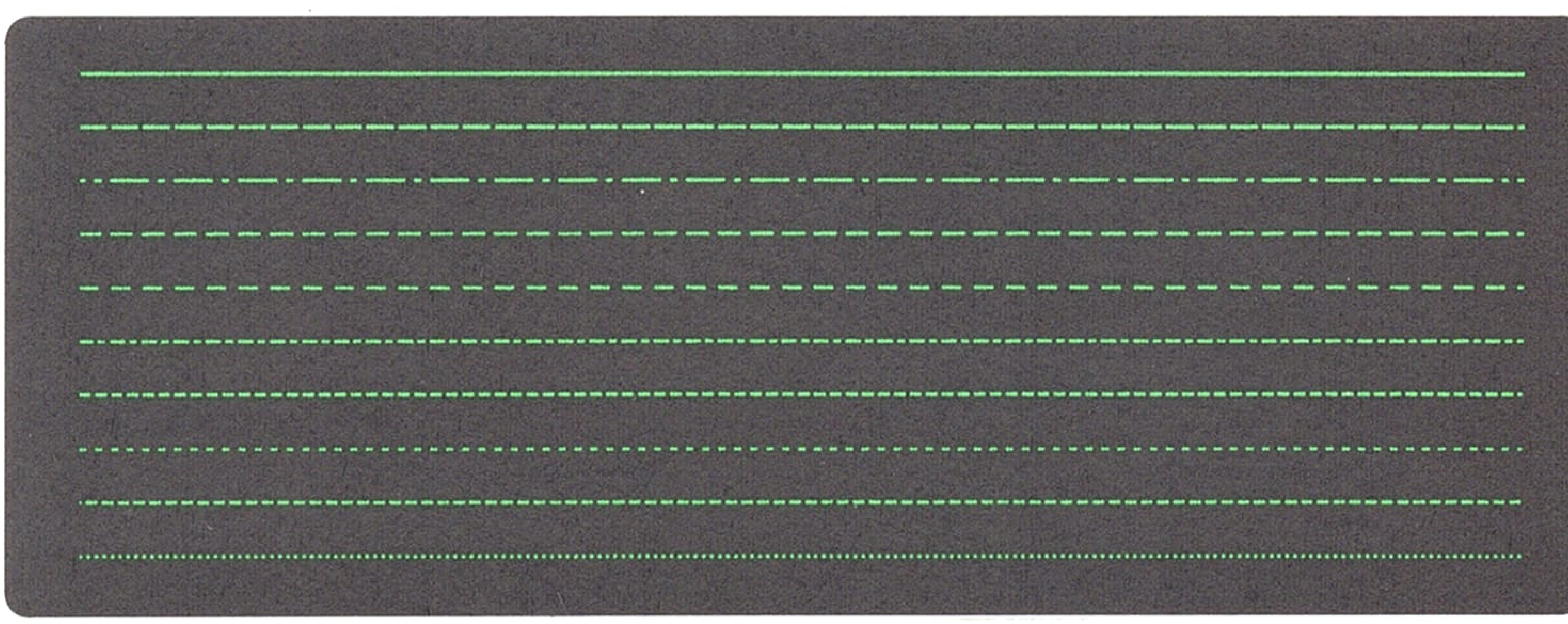
ラインスタイルはFFFFとか、FEFEとかAAAAと選んでありますが、でたらめに選んであるわけではありません。でたら目に選んだラインスタイルはあまり美しくありません。実線を引こうとか、点線を引こうとか、一点鎖線を引こうとかハッキリ決めてからデザインしませんと、本当に役立つものはできないと思います。偶然に期待するのは無理です。

次にいろいろな線を引くプログラムを示します。

●プログラム 3−1●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 '
140 LINE (0, 10)-STEP(639,0),1,,&HFFFF
150 LINE (0, 20)-STEP(639,0),1,,&HFEFE
160 LINE (0, 30)-STEP(639,0),1,,&HFFCC
170 LINE (0, 40)-STEP(639,0),1,,&HFCFC
180 LINE (0, 50)-STEP(639,0),1,,&HF8F8
190 LINE (0, 60)-STEP(639,0),1,,&HF6F6
200 LINE (0, 70)-STEP(639,0),1,,&HEEEE
210 LINE (0, 80)-STEP(639,0),1,,&HCCCC
220 LINE (0, 90)-STEP(639,0),1,,&HBBBB
230 LINE (0,100)-STEP(639,0),1,,&HAAAA
240 '
250 END
```

次は、プログラム３−１の実行結果です。



ラインスタイルを使うと美しい造形を生みだすことができます。よくプリンタの会社のマニュアルに記載されている絵を作ってみました。色をつけるともっと面白いものができるでしょう。

●プログラム 3−2●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 WINDOW (0,0)-(639,399)
125 VIEW (0,0)-(639,199)
130 FOR I=&H0 TO &HFF
140 LINE (0,I)-STEP(639,0),6,,I
150 NEXT I
160 END
```

● 紙面の関係で実行結果を載せられませんでした。各自お試しください。

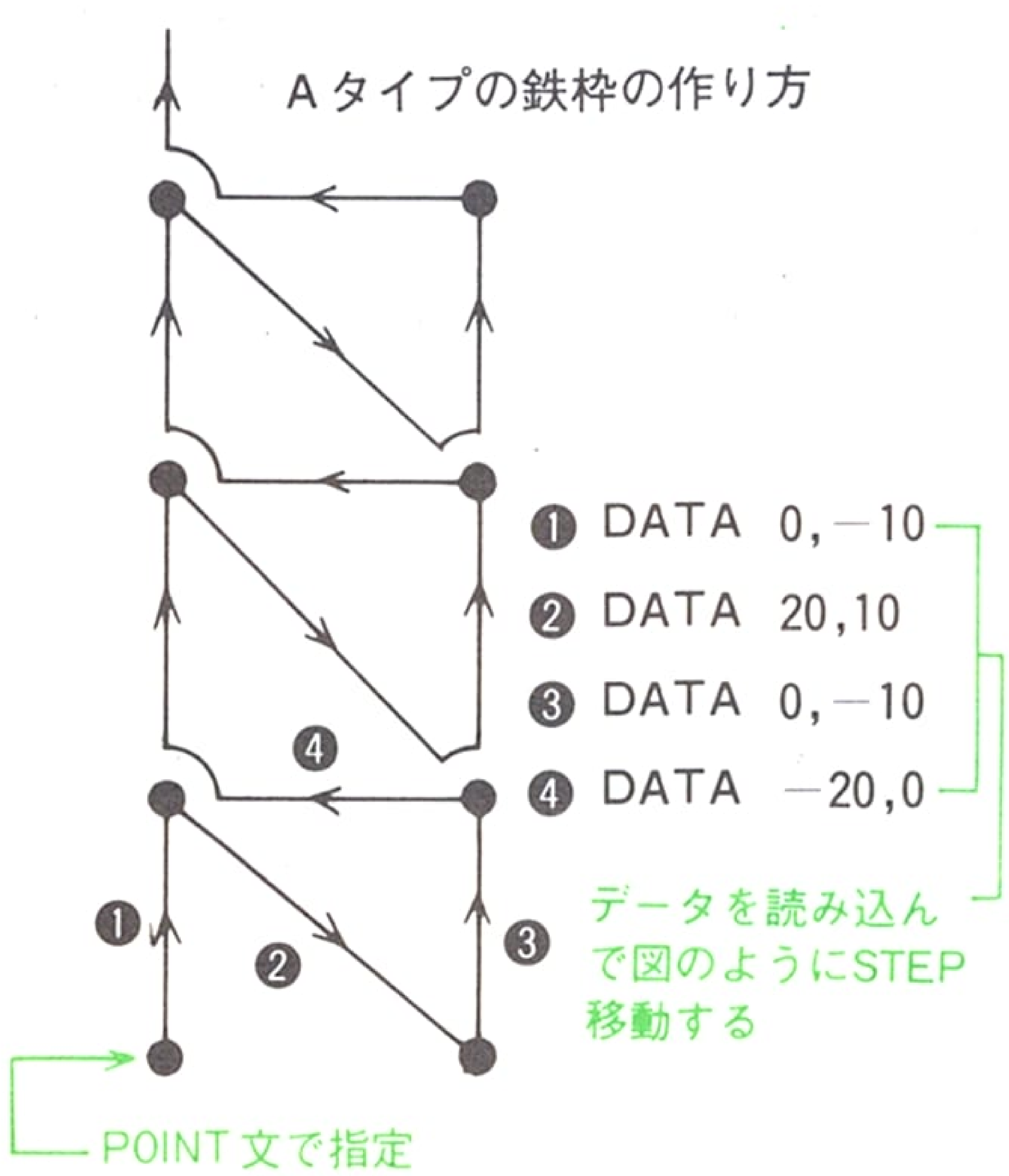
3·1 ゴールデンゲートブリッジを作る

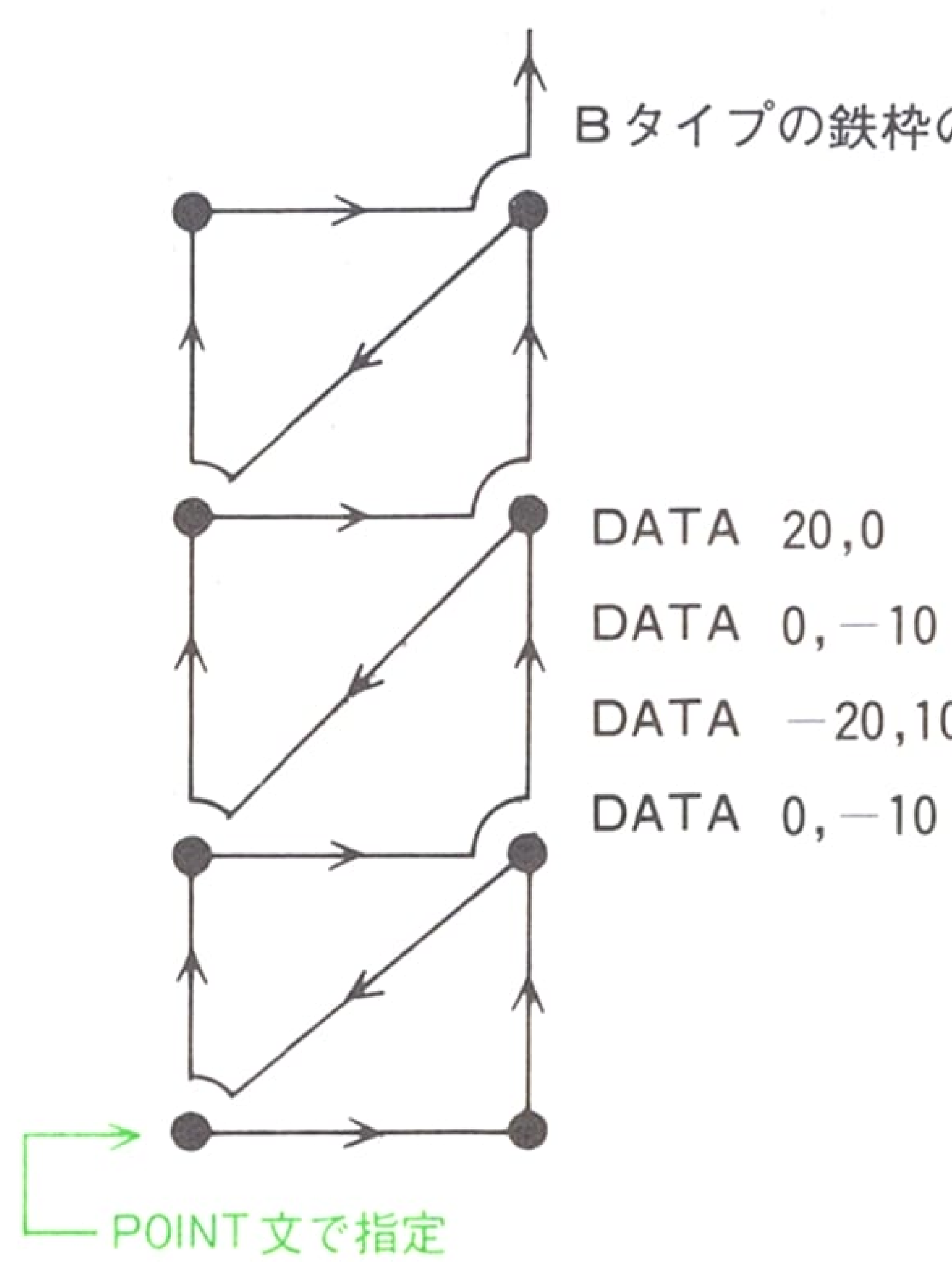
サンフランシスコに行きますとゴールデンゲートブリッジ(金門橋)という橋を見ることができます。大変美しい橋で、私も夜景を海軍基地の前から楽しんで見てきました。これを作ってみようと思いたち実行して見ました。

このプログラムを作るのには2つのポイントがありました。1つは鉄枠をどう作るかの問題と、もう1つは吊りケーブルをどう作るかの問題でした。私は吊りケーブルにラインスタイル文を使ってどれが一番リアルに見えるかを調べる例題にしようと考えました。

相当苦労して作り上げたのですが、仕上がりは思ったほどではなく、だいぶガッカリしましたが、思わぬ副産物がありましたので、お話しておきたいと思います。吊りケーブルは非常に簡単だったので、しばらく脱線して鉄枠の話をしたいと思います。

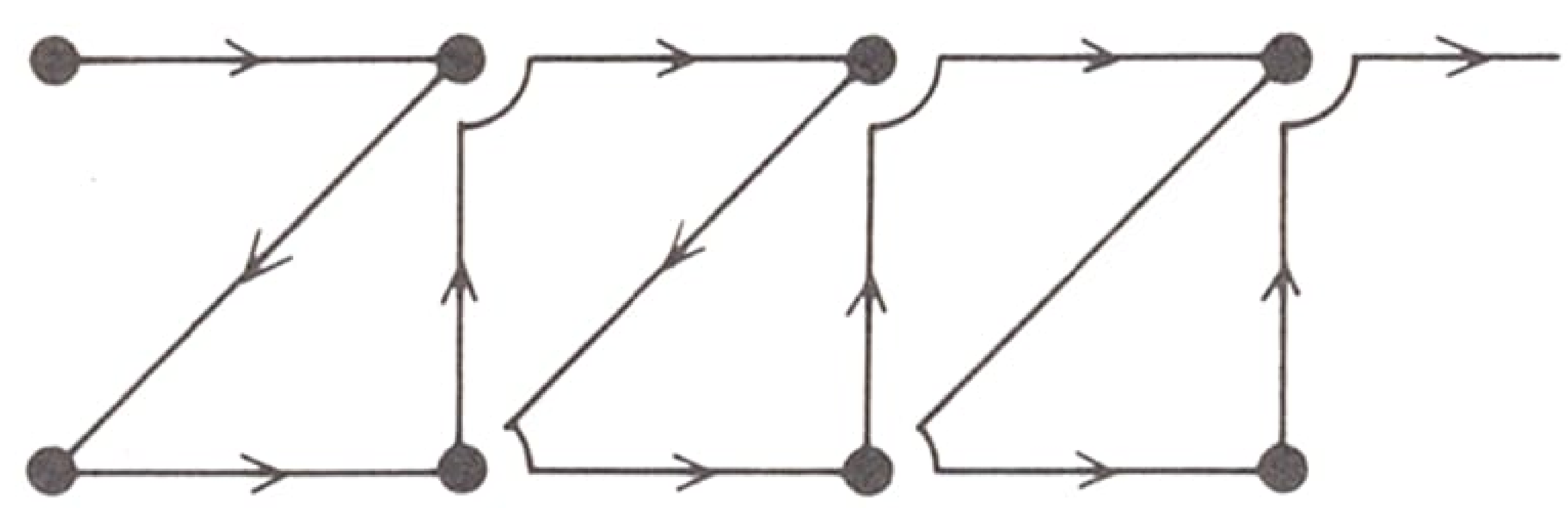
鉄枠を作るのに私は部品を積み上げて行く方式をとりたいと思いました。そこでLINE−STEP文を使うことにしました。たとえば縦方向の枠を作るのに2つのサブルーチンを作りました。仮にこれをAタイプとBタイプと呼ぶことにしましょう。Aタイプ、Bタイプは次のようにして作りました。





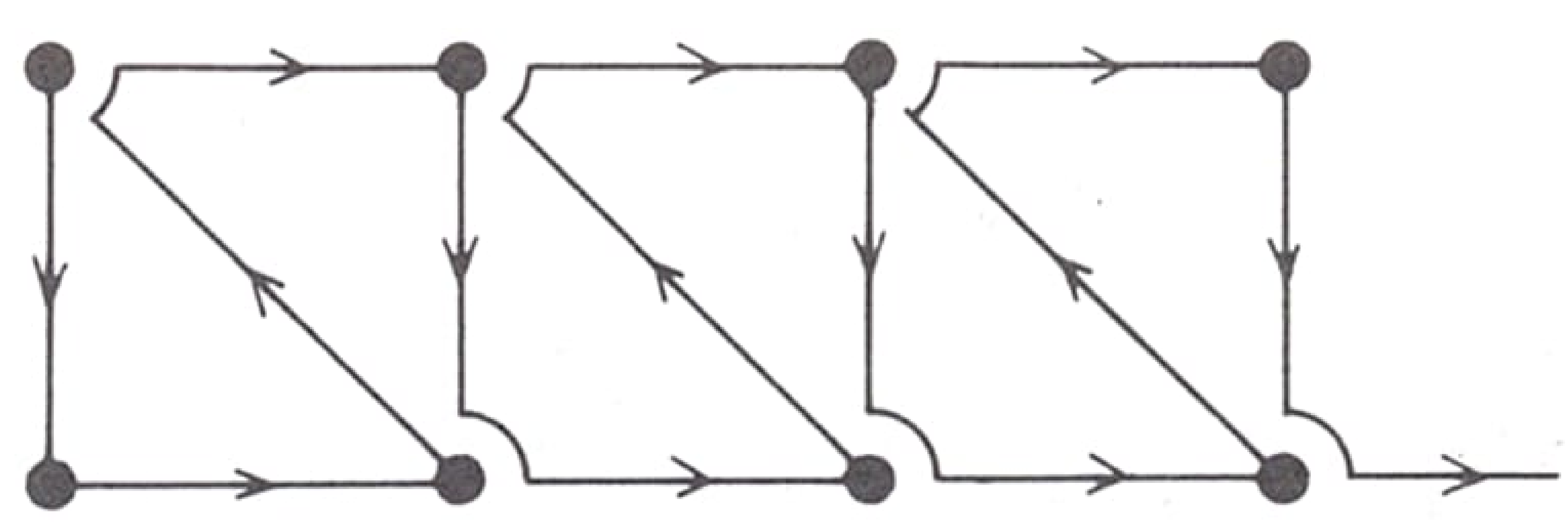
次に横方向の枠を作るのに2つのサブルーチンを作りました。これをCタイプとDタイプと呼ぶことにしましょう。

Cタイプの鉄枠の作り方



```
DATA 20,0
DATA −20,10
DATA 20,0
DATA 0,−10
```

Dタイプの鉄枠の作り方

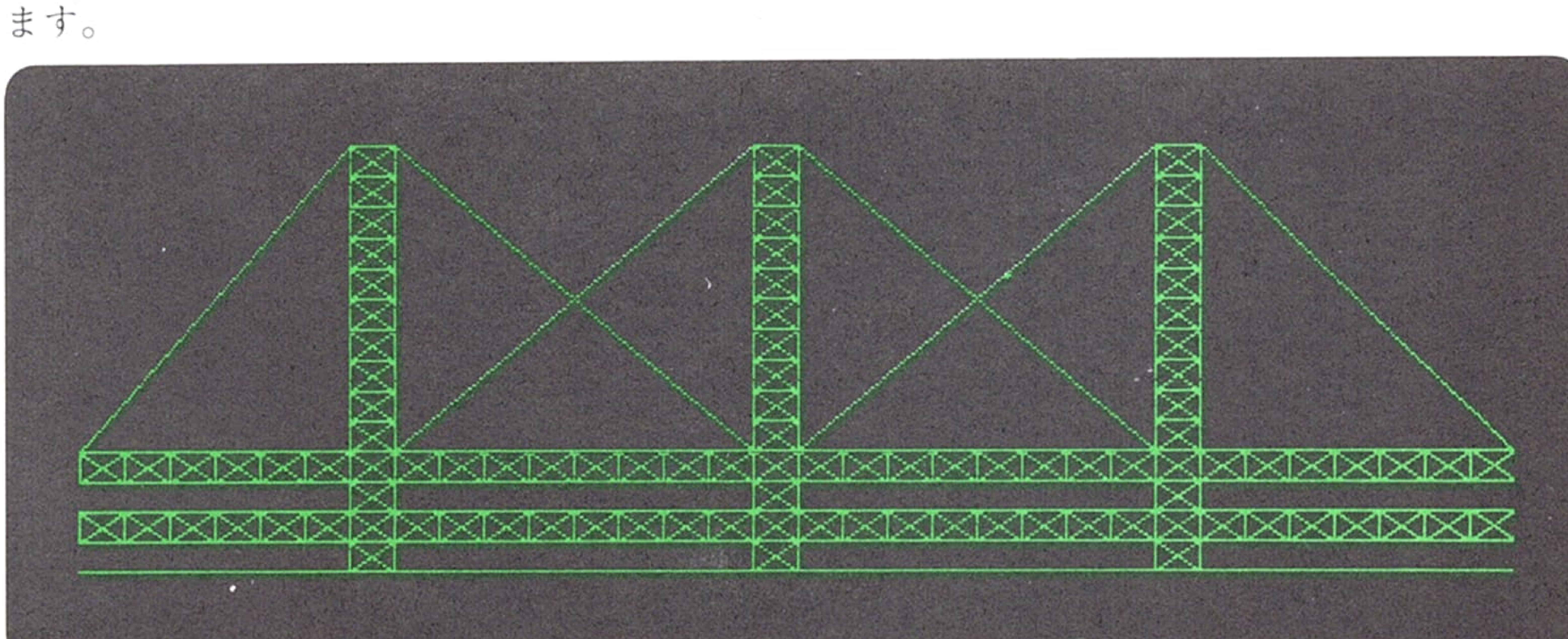


```
DATA 0,10
DATA 20,0
DATA −20,−10
DATA 20,0
```

この４つのサブルーチンを使って、だんだん増殖して行く図形を作ることにしました。かなり苦労して考えて作りましたが、絵自体はあまりパッとしないもので相当私を気落ちさせました(下図)。

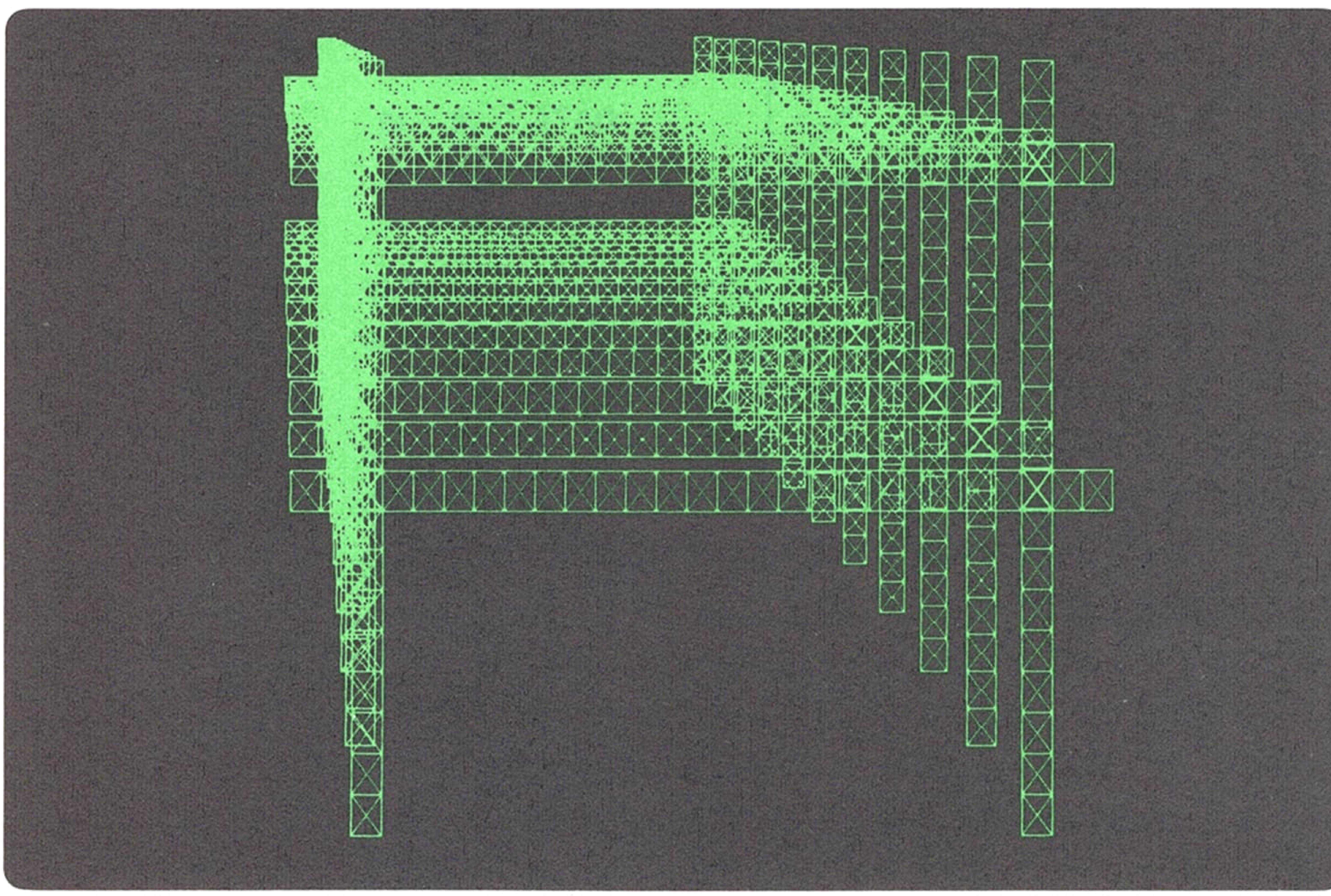
なぜ失敗したかを反省してみますと、図柄が平板的である上に、対称形の構図を採用したことが失敗の最大原因でしょう。そこで平板的にならず、非対称の構図を採用して絵を描いて見たのがプログラム３－４とその実行結果です。

WINDOWを変えているだけですが、出来上がった図形を見た時、私はアッと驚きました。何となくリアリズムがでているので本当に驚いたのです。金門橋の図と比べて下さい。これなら前のプログラムにくらべ少しは満足のできるものができそうです。コンピュータで図形を描く時のヒントはこの辺にありそうだと私は思いました。単純図形の適当なくり返しと構図のとり方の工夫が大切なようです。プログラムの長さはほとんど同じでありながら、全く違うものができてしまいます。



●プログラム 3―3の実行結果●

商品としてCFフィルムに使われているコンピュータ・グラフックスをVTRのスロー再生やコマ送り再生で徹底的に分析してみますと、この本で扱われている程度のテクニックを上手に組み合わせ、大胆に素材を簡略化して映像の世界を作り出しているようです。有名なボブエイブルの作品でも徹底的に見て行きますと、線の数は意外に少なく、使われている曲線も限られた種類のようです。むしろうまさは、少ない線で対象を鮮やかに浮かび上らせている所にあるようです。



●プログラム 3—4の実行結果

●プログラム 3—3●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 WINDOW (0,0)-(639,399):VIEW (0,0)-(639,199)
130 POINT (120,199):N=15:GOSUB 330
140 POINT (120,199):N=15:GOSUB 480
150 POINT (300,199):N=15:GOSUB 330
160 POINT (300,199):N=15:GOSUB 480
170 POINT (480,199):N=15:GOSUB 330
180 POINT (480,199):N=15:GOSUB 480
190 POINT (  0,159):N=33:GOSUB 630
```

鉄枠を作る（130〜190行）

```
200 POINT (  0,159):N=33:GOSUB 780
210 POINT (  0,179):N=33:GOSUB 630
220 POINT (  0,179):N=33:GOSUB 780
230 '
240 LINE (  0,199)-(639,199),1
250 LINE (140, 59)-(300,159),6,,&HF00F
260 LINE (320, 59)-(480,159),6,,&H3030
270 LINE (140,159)-(300, 59),6,,&HFEFE
280 LINE (320,159)-(480, 59),6,,&HFCFC
290 LINE (  0,159)-(120, 59),6,,&H5555
300 LINE (500, 59)-(639,159),6,,&HFFCC
310 END
320 '
330 S=1
340 RESTORE 430
350 READ X,Y,F
360 LINE -STEP(X,Y),2
370 IF F>0 THEN GOTO 400
380 GOTO 350
390 '
400 S=S+1:IF S=N THEN RETURN
410 RESTORE 430:GOTO 350
420 '
430 DATA        0,    -10,       0
440 DATA       20,     10,       0
450 DATA        0,    -10,       0
460 DATA      -20,      0,       1
470 '
480 S=1
490 RESTORE 580
500 READ X,Y,F
510 LINE -STEP(X,Y),2
520 IF F>0 THEN GOTO 550
530 GOTO 500
540 '
550 S=S+1:IF S=N THEN RETURN
560 RESTORE 580:GOTO 500
570 '
580 DATA       20,      0,       0
590 DATA        0,    -10,       0
600 DATA      -20,     10,       0
610 DATA        0,    -10,       1
620 '
```

海面を作る（240）

吊りケーブルを作る（250〜300）

Aタイプの鉄枠を作るサブルーチン（340〜460）

Bタイプの鉄枠を作るサブルーチン（490〜610）

```
630 S=1
640 RESTORE 730
650 READ X,Y,F
660 LINE -STEP(X,Y),2
670 IF F>0 THEN GOTO 700
680 GOTO 650
690 '
700 S=S+1:IF S=N THEN RETURN
710 RESTORE 730:GOTO 650
720 '
730 DATA        20,       0,       0
740 DATA       -20,      10,       0
750 DATA        20,       0,       0
760 DATA         0,     -10,       1
```

Cタイプの鉄枠を作るサブルーチン

```
770 '
780 S=1
790 RESTORE 880
800 READ X,Y,F
810 LINE -STEP(X,Y),2
820 IF F>0 THEN GOTO 850
830 GOTO 800
840 '
850 S=S+1:IF S=N THEN RETURN
860 RESTORE 880:GOTO 800
870 '
880 DATA         0,      10,       0
890 DATA        20,       0,       0
900 DATA       -20,     -10,       0
910 DATA        20,       0,       1
```

Dタイプの鉄枠を作るサブルーチン

●プログラム 3-4●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 WINDOW (0,0)-(639,399):VIEW (0,0)-(639,199)
130 FOR I=0 TO 10
140 WINDOW (0,0)-(639+50*I,399+50*I)
150 POINT ( 50,399):N=20:GOSUB 270
160 POINT ( 50,399):N=20:GOSUB 420
170 POINT ( 10, 60):N=28:GOSUB 570
180 POINT ( 10, 60):N=28:GOSUB 720
190 POINT ( 10,220):N=28:GOSUB 570
```

```
200 POINT ( 10,220):N=28:GOSUB 720
210 POINT (490,399):N=20:GOSUB 270
220 POINT (490,399):N=20:GOSUB 420
230 NEXT I
240 '
250 END
260 '
270 S=1
280 RESTORE 370
290 READ X,Y,F
300 LINE -STEP(X,Y),2
310 IF F>0 THEN GOTO 340
320 GOTO 290
330 '
340 S=S+1:IF S=N THEN RETURN
350 RESTORE 370:GOTO 290
360 '
370 DATA        0,      -20,      0
380 DATA       20,       20,      0
390 DATA        0,      -20,      0
400 DATA      -20,        0,      1
410 '
420 S=1
430 RESTORE 520
440 READ X,Y,F
450 LINE -STEP(X,Y),2
460 IF F>0 THEN GOTO 490
470 GOTO 440
480 '
490 S=S+1:IF S=N THEN RETURN
500 RESTORE 520:GOTO 440
510 '
520 DATA       20,        0,      0
530 DATA        0,      -20,      0
540 DATA      -20,       20,      0
550 DATA        0,      -20,      1
560 '
570 S=1
580 RESTORE 670
590 READ X,Y,F
600 LINE -STEP(X,Y),2
610 IF F>0 THEN GOTO 640
620 GOTO 590
```

```
630 '
640 S=S+1:IF S=N THEN RETURN
650 RESTORE 670:GOTO 590
660 '
670 DATA        20,        0,       0
680 DATA       -20,       20,       0
690 DATA        20,        0,       0
700 DATA         0,      -20,       1
710 '
720 S=1
730 RESTORE 820
740 READ X,Y,F
750 LINE -STEP(X,Y),2
760 IF F>0 THEN GOTO 790
770 GOTO 740
780 '
790 S=S+1:IF S=N THEN RETURN
800 RESTORE 820:GOTO 740
810 '
820 DATA         0,       20,       0
830 DATA        20,        0,       0
840 DATA       -20,      -20,       0
850 DATA        20,        0,       1
```

3·2 色を塗ること(ペイントタイル文)

私は現在のパソコンの能力でグラフィックスに色をつけろというのは酷な注文だと思っています。いろいろなグラフィックの実例を見ても、そうとう高価なコンピュータを使ったコンピュータ・グラフィックスでも色を塗ると、全くパッとしないものになってしまうようです。

現在のパソコンでは輝度変調がかからないので、同じ色でも明るい暗いの区別はだせませんし、色の微妙な味わいが全然でません。実際、色を塗ってみると、線画に色紙をはり込んだようなモノトーンな感じで色にツヤもなければ深みもなく、単調で寂しい感じしか出ないのです。これは大変困ったことです。

けれど良く考えてみますと、普通に我々が描く絵でも色を塗るのはかなり難しい作業で、デッサンがやっと仕上って色を塗ると途端につまらないものになってしまったことを思い出します。下塗りをやって色を重ねてもなかなかうまく行かず、インスピレーションのようなもので、作られた色がマッチするということがほとんどでしょう。

絵具をまぜて新しい色を作るのも大変難しい作業でした。模型に色を塗る時には一度にたくさん同じ色を作っておかないと、二度と同じ色を再現することはできませんでした。ジャーマン・グレイとか、オリーブ・グリーンとかサンド・イエローなどという色は素人には絶対に同じ色が作れないので有名です。

電磁気学の基本方程式であるMaxwellの方程式を発見したジェムズ・クラーク・マクスウェルは電磁気学の大家であるだけでなく、統計力学のMaxwell分布とか、土星の輪は小さなイン石の集合体であることを証明したり、楕円の描き方を発見したり、犬の眼の中をのぞく眼鏡を作ったりした他に、光学の大家でもありました。実はマクスウェル(最近の学生は嘆かわしくもMaxwellをMaxellと書きますが、あれは日立のブランドの1つです)においては、これらのことは1つの体系の一部で、この天才は力学的自然観にすぐれた人でした。

もう少しハッキリ言いますと、マクスウェルの得意としたのは機械や歯車やネジやコマで、特にコマにすぐれていました。マクスウェルの『悪魔』というコマは特に有名です。土星の輪は小さなイン石の集合体であるという証明の中でも福引の機械のようなガラガラ車がでてきたり、Maxwellの方程式も、そもそもは流体力学と歯車のアナロジーから出発しています。

●統計力学のマクスウェルの悪魔とは違います。マクスウェルは悪魔が大好きで色々な悪魔を発明しました。

話が少し脱線しましたが、マクスウェルはコマにすぐれており、光

学にもすぐれていたので、マクスウェルの色ゴマというものを作りました。これはどういう色を混ぜると、どういう色ができるかを教えてくれるもので原理は簡単ですが、塗料屋さんでは、つい最近まで使われていたと聞きます。この色ゴマによって始めて色の再現性が高まったというくらい、色を作ることは悩ましいことなのです。だから最初のうちは色をうまく扱えなくとも仕方がないのかも知れません。このようにいろいろ留保を付け予防線を張るのは、一見非常に強力そうに見えるPAINT TILE$文もオールマイティではないということを強調したいからです。

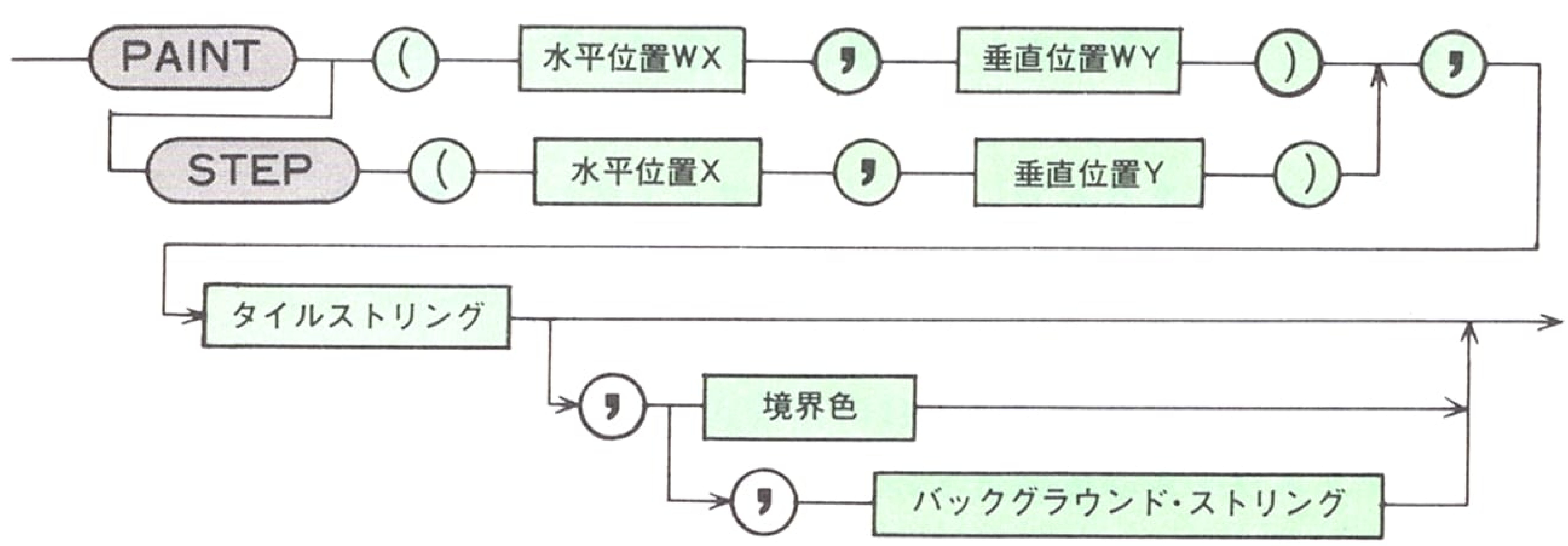
まずPAINT TILE$文の構造図をかかげておきましょう。

PAINT ●paint／ペイント

目　的　指定された境界色で囲まれた領域を，指定されたタイルパターンで埋める

書　式　PAINT　(WX,WY)　,〈タイルストリング〉〔,〈境界色〉〕
　　　　STEP(X,Y)
　　　　〔,〈バックグラウンド・ストリング〉〕

文　例　PAINT (255,120),TILE$,4,BACK$



タイルはラインスタイル文と考え方は同じです。違うのは1次元的な線が2次元的な領域になったということです。タイルの指定はモノクロモードとカラーモードでは違います。モノクロの場合にはラインスタイル文と同じように使えば良いのですが、カラーモードの場合には青、赤、緑について別々にタイルを指定してやらなければならない所が変わっています。タイルはCHR$（★）を使って指定することになります。CHR$（★）の★の部分は&H＋16進数で表示して下さい。

モノクロモードの場合、★の16進数を2進数のビットパターンで表現した場合、1のビットに対応するドットがセット、0のビットに対応するドットがリセットされます。わからない方はもう一度ラインスタイルのところをご覧ください。

カラーモードの場合も考え方は同じですが、プレーンが青、赤、緑と3つあるため、3つのタイルストリングを指定しなければなりません。

最初に、8つの円を黒、青、赤、紫、緑、水色、黄、白と塗りつぶすプログラムを作ってみました。

●プログラム 3−5●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 WINDOW (0,0)-(639,399):VIEW (0,0)-(639,199)
130 FOR I=0 TO 7
140 CIRCLE(40+80*I,100),35,7
150 NEXT I
160 '
170 PAINT ( 40,100),CHR$(&H0)+CHR$(&H0)+CHR$(&H0),7
180 PAINT (120,100),CHR$(&HFF)+CHR$(&H0)+CHR$(&H0),7
190 PAINT (200,100),CHR$(&H0)+CHR$(&HFF)+CHR$(&H0),7
200 PAINT (280,100),CHR$(&H0)+CHR$(&H0)+CHR$(&HFF),7
210 PAINT (360,100),CHR$(&HFF)+CHR$(&HFF)+CHR$(&H0),7
220 PAINT (440,100),CHR$(&HFF)+CHR$(&H0)+CHR$(&HFF),7
230 PAINT (520,100),CHR$(&H0)+CHR$(&HFF)+CHR$(&HFF),7
240 PAINT (600,100),CHR$(&HFF)+CHR$(&HFF)+CHR$(&HFF),7
250 '
260 END
```

注記：
- 100〜110行：カラーモード
- 130〜150行：7つの白い円を描く
- 170行：この場合、どのプレーンもデータが0なので色がでない
- プレーン1（青）、プレーン2（赤）、プレーン3（緑）

プログラム3−5はわかりやすくて良いのですがCHR$をたくさん使わされるのでファンクション・キーにでも登録していない限り、入力に手間がかかってかないません。そこでもう少し見やすく、わかりやすいプログラムを作ってみました。DATA文を使っていますので、この方が楽になります。プログラムの実行結果は前のプログラムと全く同じになります。

●プログラム 3−6●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 WINDOW (0,0)-(639,399):VIEW (0,0)-(639,199)
130 FOR I=0 TO 7
140 CIRCLE(40+80*I,100),35,7
150 NEXT I
160 '
170 FOR I=1 TO 7
180 TILE$=" "
190 READ A$,B$,C$
200 TA=VAL("&H"+A$)
```

```
210 TB=VAL("&H"+B$)
220 TC=VAL("&H"+C$)
230 TILE$=CHR$(TA)+CHR$(TB)+CHR$(TC)
240 PAINT(40+80*I,100),TILE$,7
250 NEXT I
260 '
270 DATA FF,00,00,00,FF,00,00,00,FF,FF,FF,00
280 DATA FF,00,FF,00,FF,FF,FF,FF,FF
```

次にデータを変えて、8色以外の色を作ってみました。今度は中間色がでてきます。中間色の作り方は無数といって良いほどあるので、あらかじめどんな色がでてくるかは、ここでのプログラムの方式ではわかりませんが、データを変更すれば、どんどん新しい中間色が作れます。

ただし批判的に見た場合、この方式にはどんな色ができるか、あらかじめ了解することができないという欠点があります。色を作るにはあらかじめ欲しい色が、どの位の比率で3原色をまぜれば良いかがわかっていなければなりませんし、3原色の比率がわかっている場合でも、DATA文をどう作れば良いかがわかっていないと困るのです。

☞ 4 (P.214)

●プログラム 3—7●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 WINDOW (0,0)-(639,399):VIEW (0,0)-(639,199)
130 FOR I=0 TO 7
140 CIRCLE(40+80*I,100),35,7
150 NEXT I
160 '
170 FOR I=1 TO 7
180 TILE$=" "
190 READ A$,B$,C$
200 TA=VAL("&H"+A$)
210 TB=VAL("&H"+B$)
220 TC=VAL("&H"+C$)
230 TILE$=CHR$(TA)+CHR$(TB)+CHR$(TC)
240 PAINT(40+80*I,100),TILE$,7
250 NEXT I
260 '
270 DATA AA,55,00,00,AA,55,55,00,AA,FF,FF,00
280 DATA FF,00,FF,00,FF,FF,FF,FF,FF
```

3・3 タイルパターンを作るためのツール

色は不可思議玄妙なものです。思うような色はデタラメでは絶対にできません。とは言え中間色の構成のために色のドットパターンで作り、それを16進数に直すのを手作業でやったのではたまりません。何とか楽にやる方法はないでしょうか？　我々は PC-８８０１ という強力なパソコンを持っているのですから、かなりの作業を手伝ってもらいたいものです。

それは十分可能なことです。これまでにもいろいろ巧みなプログラムが発表されていますが、私も１つ作ってみました。あまり長いプログラムですと、入力が大変なので機能を欲ばらないことにしました。なるべくテキスト画面とグラフィック画面を分離して、テキスト画面に負担をかけないようにしました。もう少し工夫するともっと使いやすくなったとは思いますが、長くなるのを避けたいと思い、断念しました。使って見たところでは特に支障はなかったと思います。

プログラムはメインプログラムと３つのサブルーチンでできています。簡単に説明しておきましょう。

＊MAKETILE

タイルをグラフィック画面に作ります。カラーコードだけを入力すれば、次のマス目に移動するようにしてあります。

＊TRAP

入力を間違えた時 (f･1) キーを押すと自動的に処理を中断し、最初からやりなおすようになっています。

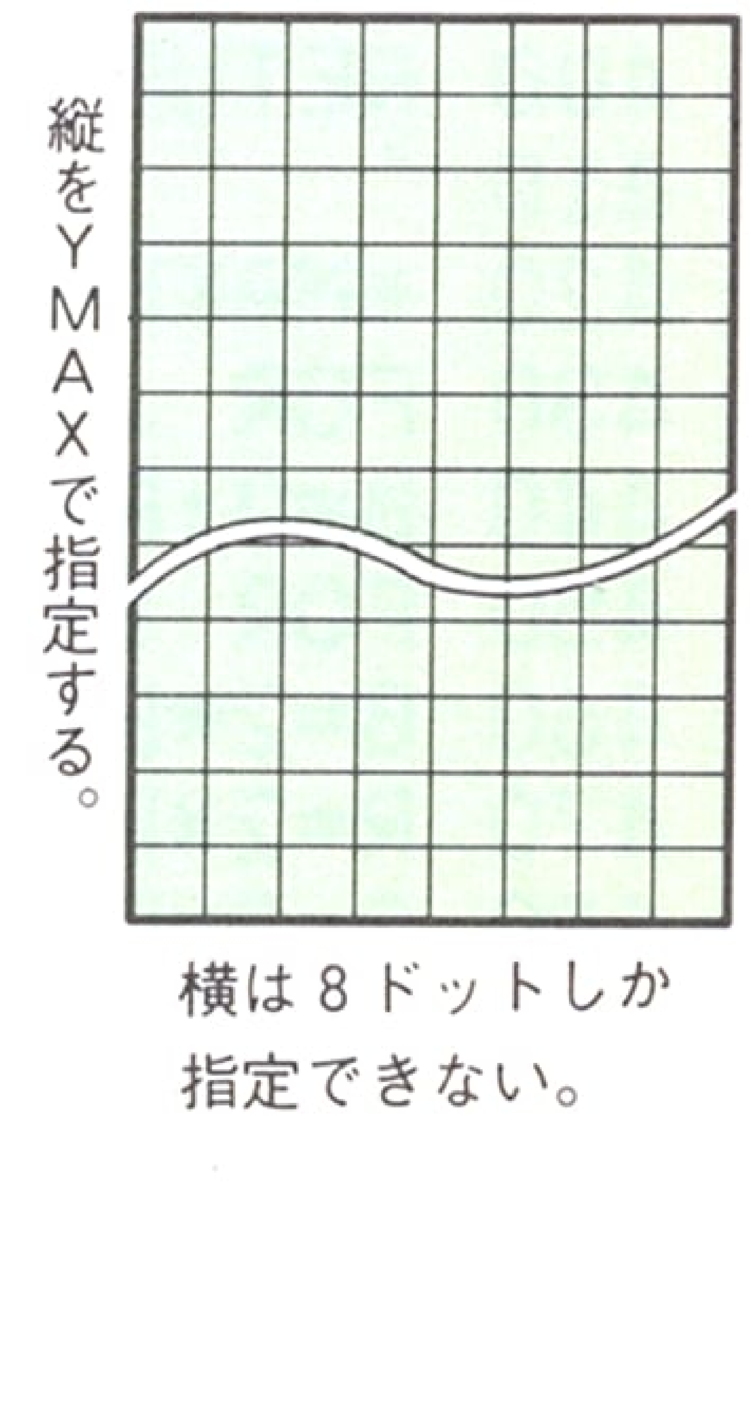
＊MAKEDATA

TILE$ 文を作っています。

＊PRINTDATA

画面に DATA 文を打ち出します。

プログラムを動かすと最初に「YMAX?」と聞いてくるはずです。カラーの場合 TILE$ 文は横８ドットしか指定できないので、自由度があるのは縦方向だけです。縦方向のドット数を YMAX で指定します。一応25まで指定するように作ってありますが、あまりたくさんとっても意味がないように思います。意味のあるパターンは YMAX が16 ぐらいまででしょう。



横は８ドットしか
指定できない。

●プログラム 3－8●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 DIM T(8,25):DEFINT B,R,G          ← 配列設定と整数宣言
130 ON KEY GOSUB *TRAP                ← 入力ミスの時ファンクションキー1を押せば
140 KEY (1) ON                           *TRAPへジャンプする
150 LOCATE 40,0:COLOR 6
160 INPUT"YMAX     ";Y                ← 縦方向のドット数入力
170 IF Y<0 OR Y>25 THEN GOTO 150
180 GOSUB *MAKETILE                   ← タイルを描く
190 GOSUB *MAKEDATA                   ← 各タイルのデータを加算してTILESをつくる
200 GOSUB *PRINTDATA                  ← データの表示
210 END
220 '
230 *MAKETILE
240 FOR J=1 TO  Y
250 FOR I=1 TO  8
260 POINT(20*I,10*J)                  ← タイルの左上の端点
270 LOCATE 40,1:COLOR 5               ← カラーコードの入力とその塗られるタイル
280 PRINT "X=";I;"Y=";J                  の位置表示
290 LOCATE 40,2:COLOR 5
300 INPUT"COLOR    ";COL
310 IF COL<0 OR COL>7 THEN GOTO 270
320 T(I,J)=COL                        ← 入力されたカラーコードを配列に代入
330 LINE STEP(0,0)-STEP(20,10),COL,BF
340 NEXT I
350 NEXT J
360 RETURN
370 '
380 *TRAP
390 COLOR 2:CLS:LOCATE 40,0           ← 入力ミスの時、最初からやりなおす
400 RETURN 160
410 '
420 *MAKEDATA
430 FOR J=1 TO  Y
440 B=0:R=0:G=0                       ← 変数(BLUE、RED、GREEN)をイニシャライズ
450 FOR I=1 TO  8                     ← 1、2、4、つまり青、赤、緑のカラーコードと
460 B=2*B+(T(I,J) AND 1)¥1               1行分8つのタルカラーコードとの論理
470 R=2*R+(T(I,J) AND 2)¥2               演算をして3つのプレーンの値を決める
480 G=2*G+(T(I,J) AND 4)¥4               (カラー分解をしている)
490 NEXT I
500 T$=T$+CHR$(B)+CHR$(R)+CHR$(G)     ← タイルストリングを縦方向へ加算
                                         (CHR$(B)+CHR$(R)+CHR$(G): 横8ドットのタイルストリング)
510 NEXT J
```

```
520 RETURN
530 '
540 *PRINTDATA
550 C=0
560 FOR J=1 TO Y
570 LOCATE 40,10+J
580 PRINT "DATA  ";
590 A$="0"+HEX$(ASC(MID$(T$,C+1,1)))
600 PRINT RIGHT$(A$,2);"  ";
610 C=C+1
620 IF (C MOD 3)<>0 THEN GOTO 590
630 PRINT
640 NEXT J
650 RETURN
```

カウンタの初期化（550行）

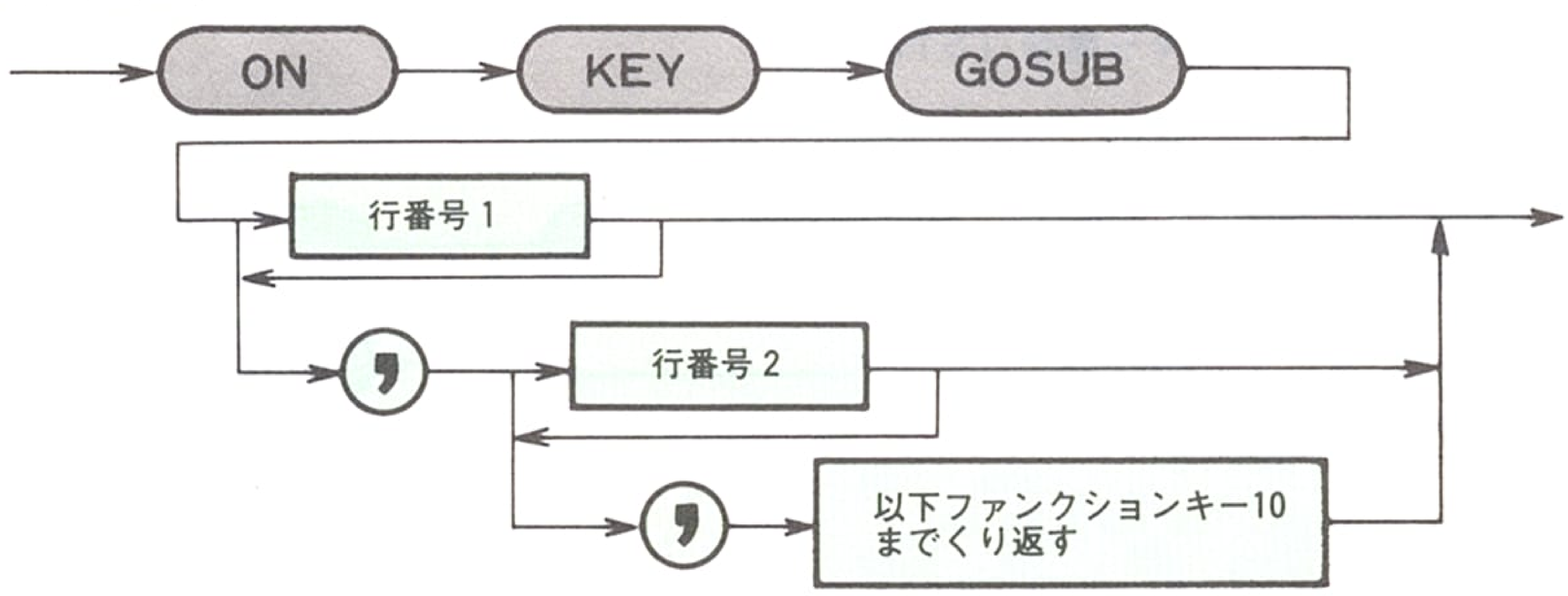
青、赤、緑のタイルストリングをデータとして表示（590〜620行）

ON KEY GOSUB ●オン キー ゴートゥ サブルーチン

目 的 ファンクションキーによる割り込みルーチンの開始行を定義する

書 式 ON KEY GOSUB 〈行番号〉〔,〈行番号〉…〕

文 例 ON KEY GOSUB 100,200



この命令は、割り込みが発生した時（キーが押された時）にどの処理ルーチンに飛ぶかを定義するだけのものであり、この命令が実行されても割り込みが起こることはありません。割り込みが発生するのは、あくまで何らかの外的要因（この場合、キーが押されたこと）があった時です。

〈行番号〉とは、割り込みが発生した時に分岐させる処理ルーチンの開始行番号で、この並びの順序は、ファンクションキーのキー番号と１対１に対応していなければなりません。PC-８８０１がサポートしているファンクションキーは全部で10番までありますから、最大10個の〈行番号〉を並べて書くことができます。もちろん実際の行番号の代わりにラベル名を使うこともできます。

処理ルーチンからの復帰は、一般のサブルーチンの場合と同じで RETURN 命令によって行います。単に RETURN とした時は、中断した所から再開し、後ろに行番号を指定した時、その行から再開します。

注意 この命令によって割り込み処理ルーチンに分岐させるには、割り込みが発生した時、KEY ON の状態でなければなりません。

（NEC PC-８８０１レファレンスマニュアル）

このプログラムでは配列Tは次のようになっています。

	I＝1	2	3	4	5	6	7	8
J＝1	(1,1)	(2,1)	(3,1)	(4,1)	(5,1)	(6,1)	(7,1)	(8,1)
2	(1,2)	(2,2)	(3,2)	(4,2)				
3	(1,3)	(2,3)	(3,3)					
4	(1,4)	(2,4)						
5	(1,5)							
6〜Y	(1,6)							(8,6)

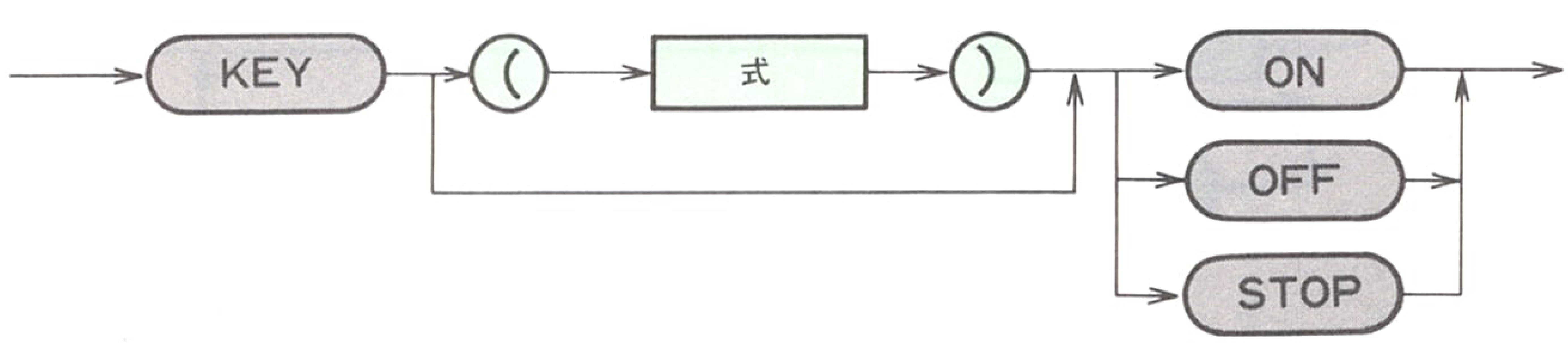
←たとえばこの行が、
赤青青赤緑緑赤黄
という表示になっている時、一番左の赤を例にとると色コードは 2ですから2進表示は、
赤──→0010
1──→0001
このANDをとると0000になります。従って、
(T (1,1) AND1) ¥ 1＝0
つまり、B＝2✳0＋0＝0
同様に、赤──→0010
2──→0010
このANDは0010、つまり2となります。従って、
(T (1,1) AND2) ¥ 2＝1
R＝2✳0＋1＝1
460—480行ではこのようにして青赤緑のプレーンのタイルストリングを決めていきます。

KEY(n) ON/OFF/STOP ●キーオン/オフ/ストップ

目的 ファンクションキーによる割り込みの許可、禁止、停止を設定する

書式
①KEY〔〈キー番号〉〕ON
②KEY〔〈キー番号〉〕OFF
③KEY〔〈キー番号〉〕STOP

文例 KEY ON



ファンクションキーが押された時に起こる割り込みを許可するか (ON),禁止するか (OFF),停止するか (STOP) を設定します。

〈キー番号〉とは、1から10までのプログラマブル・ファンクションキーの番号を表します。省略された場合は1から10までのすべてのファンクションキーに対して実行します。

❶ 割り込み動作を許可します。この命令実行後は、指定された〈キー番号〉のファンクションキーを押すごとに割り込みを発生し、ON KEY GOSUB によって定義された処理ルーチンに分岐します。

❷ 割り込み動作を禁止します。この命令実行後は、ファンクションキーを押しても処理ルーチンへの分岐は起こりません。

❸ 割り込み動作を停止します。この命令実行後は、ファンクションキーを押しても押されたことを覚えているだけで、処理ルーチンへの分岐は起こりません。しかし、その後KEY(n) ON命令によって割り込みか許可されると、先程のファンクションキーを押したことによって処理ルーチンに分岐します。

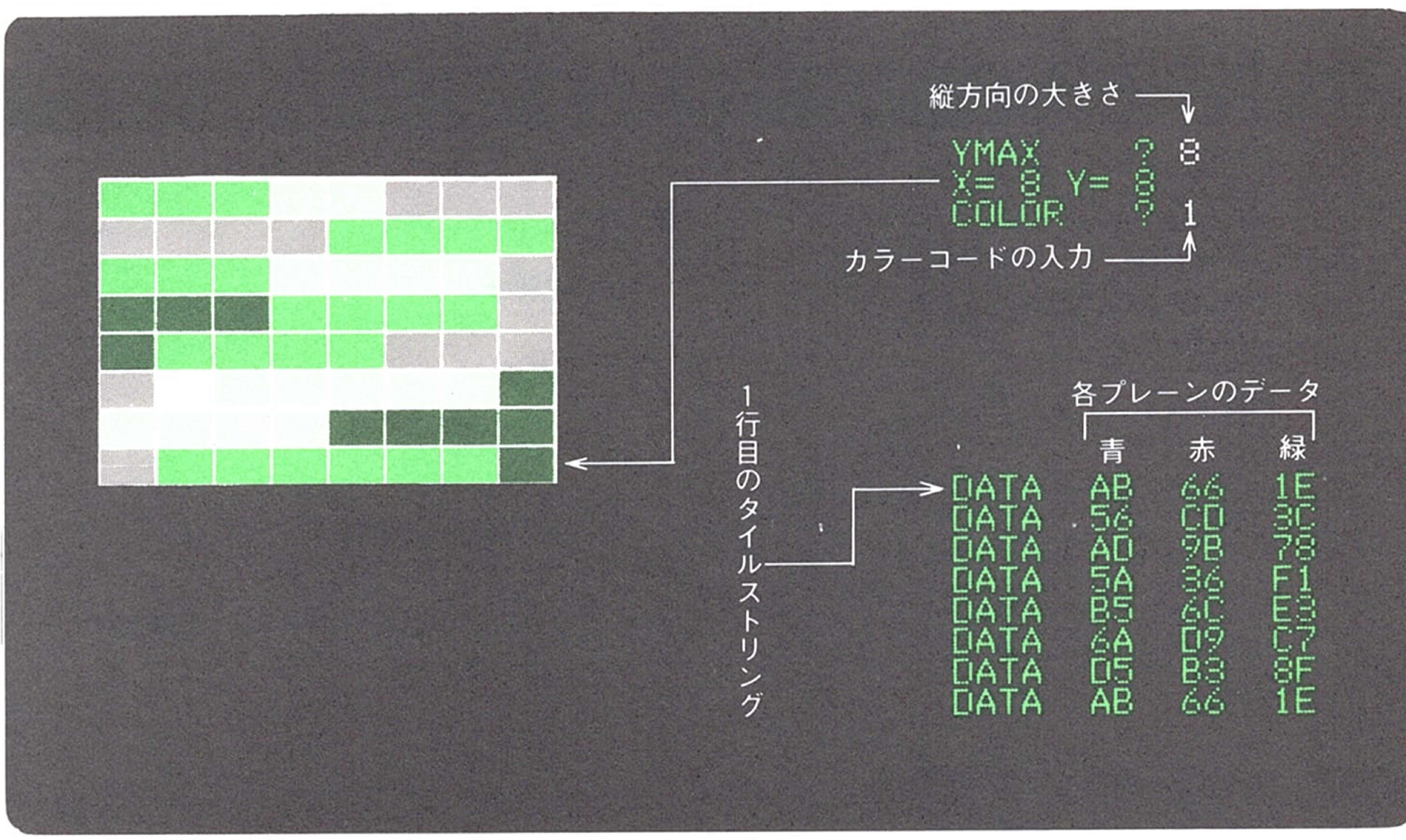
(NEC PC-8801レファレンスマニュアル)

実際にプログラムを動かした状況を図示し、プログラムの使い方について説明しておきましょう。

☞ 5 (P.215)

プログラムをRUNさせると「YMAX?」と聞いてきますから、自分で作ろうとする縦方向の数を入力します。ここでは8と選んでいます。すると画面に座標が示され、「COLOR?」と聞いてきますので、1とか2とか0から7までの数字を入力してやります。すると対応するブロックがその色で塗られるはずです。

図にある枠は実際には表示されません。枠が入ると別の色に見えてくることがあるのではずしてあります。全部のマス目に色を入力するとDATA文が打ち出されてきます。このままDATA文として使いたい時は、カーソルを動かして各データの間にカンマ(,)を入れ、「DATA」という表示の前に適当な行番号をキーインし、(RETURN)キーを押しますと、そのままDATA文として使えます。



このプログラムで赤と黒のダンダラを作ってみると、次のようになります。3行は余分で2行で良かったことに後で気がつきました。



　このデータを使って円に色を塗ってみましょう。新しくプログラム３－９を準備します。単純な赤ではなく、赤地に黒が入っているので、少し強い感じの赤になるはずです。いわゆる赤と新しく作った赤を比較してみますと、強い赤になりますが残念ながら期待したほどの効果は出ませんでした。ドットが見えてしまうからですし、もともとの赤が朱色に近いせいもあるからだと思います。

　色を作ることが難しいものだと教えてくれた例がもう１つあります。タイルパターンを作るプログラムで赤と白をまぜて見ようと思いました。赤と白を交互に表示すれば、ピンクになるはずです。DATAを作ってプログラムを走らせてみました。ピンクどころか、白っぽいベージュのお化けができてしまいました。これは赤が赤でなくて、朱色に近いからです。画面の左から赤、白、合成色と表示するようにしてありますから、是非ためして見て下さい（プログラム 3－10）。

●プログラム 3－9●

```
100 FOR I=1 TO 6
110 READ A$                  ← データの読み込み
120 TP=VAL("&H"+A$)        ┐
130 T$=T$+CHR$(TP)         ┘ タイルストリングの結合
140 NEXT I
150 PRINT T$
160 '
170 CIRCLE(420,100),50,7   ┐
180 PAINT (420,100),2,7    ┘ 画面右側に赤色の円を描く
190 CIRCLE(220,100),50,7   ┐
200 PAINT (220,100),T$     ┘ その左側に赤と黒のビットパターンで円を描く
210 END
220 '
230 DATA 00,FF,00
240 DATA 00,55,00
```

●プログラム 3－10●

```
100 FOR I=1 TO 6
110 READ A$
120 TP=VAL("&H"+A$)
130 T$=T$+CHR$(TP)
140 NEXT I
150 PRINT T$
160 '
170 CIRCLE(220,100),50,7
```

```
180 PAINT (220,100),2,7
190 CIRCLE(320,100),50,7
200 PAINT (320,100),7,7
210 CIRCLE(420,100),50,7
220 PAINT (420,100),T$
230 END
240 '
250 DATA AA,FF,AA
260 DATA 55,FF,55
```

ここで、補足説明をしておきましょう。

まず、プログラム3—10のデータをとりだしてみます。

まず行番号250のＡＡ、ＦＦ、ＡＡ、これは１行目のタイルストリングです。２進数で次のように表現できます。

```
AA   10101010  ←青のプレーン
FF   11111111  ←赤のプレーン   これを合成すると
AA   10101010  ←緑のプレーン
     ↓↓↓↓↓↓↓↓
     白赤白赤白赤白赤  ←  色コードは23ページを参照してください。
```

同じように行番号260のデータ、つまり縦方向２行目のタイルストリングですが、次のようになります。

```
55   01010101  ←青のプレーン
FF   11111111  ←赤のプレーン   これを合成すると
55   01010101  ←緑のプレーン
     ↓↓↓↓↓↓↓↓
     赤白赤白赤白赤白  ←  色コードは　ページを参照してください。
```

従って、行番号220のPAINT TILE$文ではディスプレイ上の各ドットが次のようにプロットされるはずです。

```
白 赤 白 赤 白 赤 白 赤
赤 白 赤 白 赤 白 赤 白
白 赤 白 赤 白 赤 白 赤
```

赤と白がこのように交互に分布すれば、うまくピンク色になると思ったわけです。ところが理想どおりには行きませんでした。

これで理屈がご理解いただけたと思います。

とは言え、合成した色が全くだめかと言うと、そうでもありません。赤と紫をまぜてみました(**プログラム 3—11**)。この場合は新しい美しい色が出ました。

また、青と緑の合成の場合もためしてみました(**プログラム 3—11のデータ文を変更**)。

なかなか良い色がでないのは、まだパソコンが色さえつけばカラーであるという段階にあるからでしょう。本式のコンピュータ・グラフィックスでも原色はかなり使われていても、深みのある色や味わいのある色はなかなか出せないようです。

●プログラム 3—11●

```
100 FOR I=1 TO 6
110 READ A$
120 TP=VAL("&H"+A$)
130 T$=T$+CHR$(TP)
140 NEXT I
150 PRINT T$
160 '
170 CIRCLE(220,100),50,7
180 PAINT (220,100),2,7
190 CIRCLE(320,100),50,7
200 PAINT (320,100),3,7
210 CIRCLE(420,100),50,7
220 PAINT (420,100),T$
230 END
240 '
250 DATA 55,FF,00
260 DATA AA,FF,00
```

青と緑の場合にはここを次のようにかえてください

```
DATA FF,00,55
DATA FF,00,AA
```

3・4 ペイント文で色を塗る

理屈の多い話ばかり続きましたので、ここで息抜きをしておきましょう。PAINT TILE$文を使って実際に色を塗ってみることにしましょう。あまり簡単な形は面白みがないので、すこしこったものを作ることにします。後の章でも使いたいと思いますので、題材には頭をひねりました。

対象は車体部分と上部構造物の2つからできています。2つに分けているのは後の章で上部構造物だけをぐるぐる回してみたいと思っているからです。できる限り線の数をへらしていますが、それでもかなりのデータ量になってしまいました。私のディジタイザーの使い方があまりうまくないので、データに少しひずんでいる所がありますので、美学的完成にうるさい方は適当に修正して下さい。画面とニラメッコをすれば簡単に直ります。ここにかかげるデータも本当はそうした修正処理を施してあります。

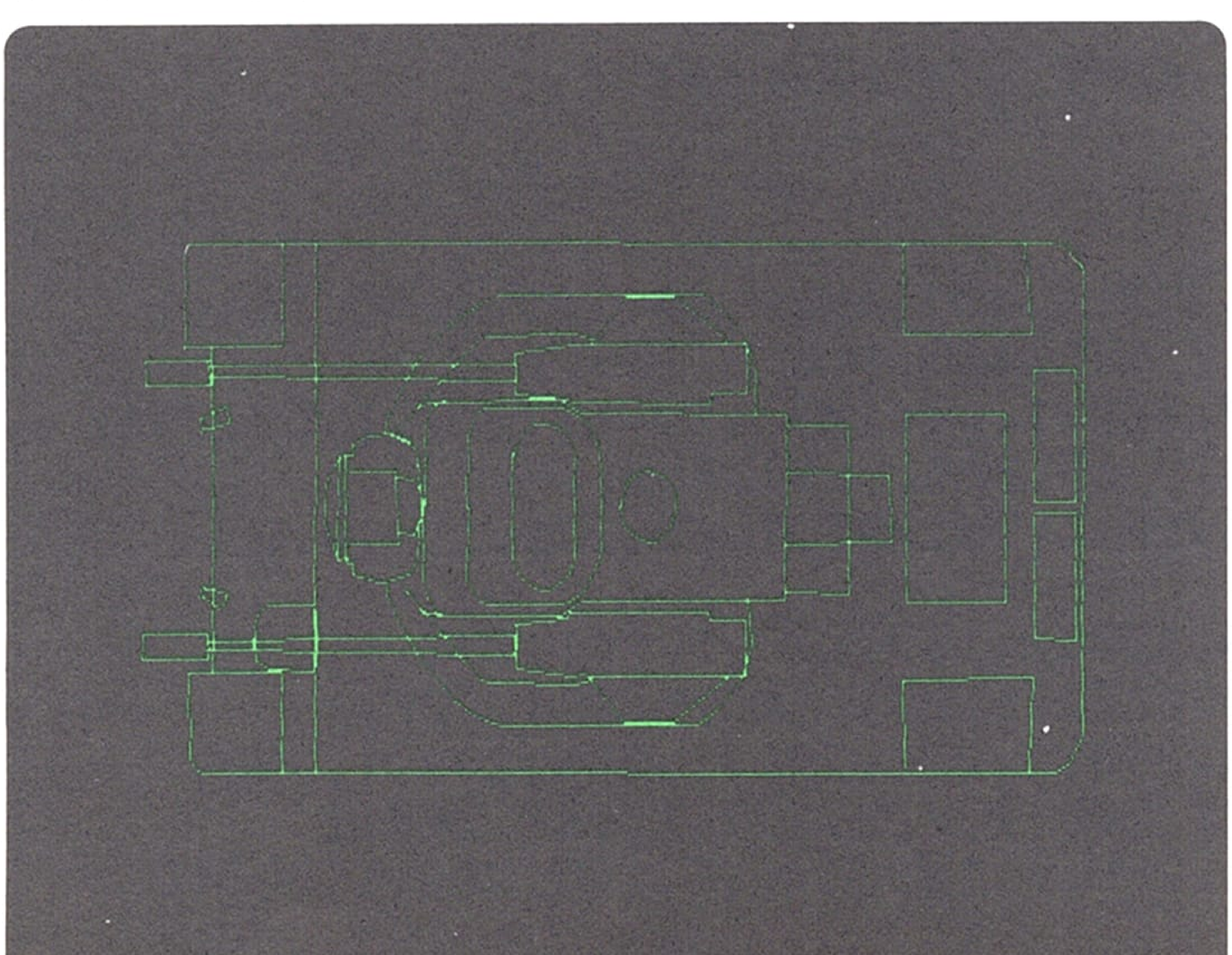
プログラムおよびデータを後ろにかかげます。

☞ 6 (P.217)

DATA 1は車体部分、DATA 2は上部構造物、DATA 3は色を塗る位置のデータです。それぞれのDATA文の終りには必ず0,0,0 をつけておいてください。

プログラムは簡単なものです。

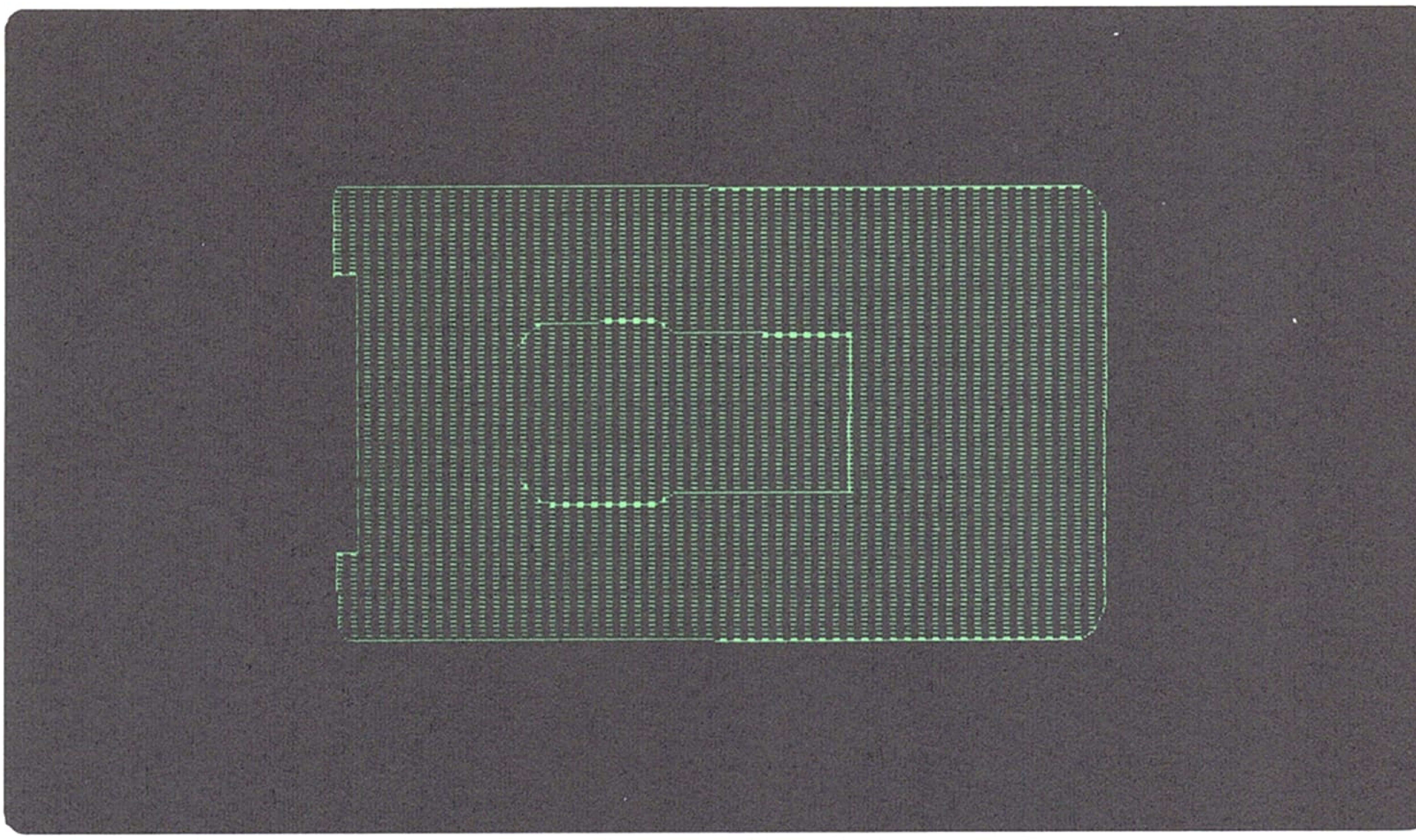
形を描いて色を塗ることだけに力をおいていますので、SCREENを特に分けるようなことはしていません。

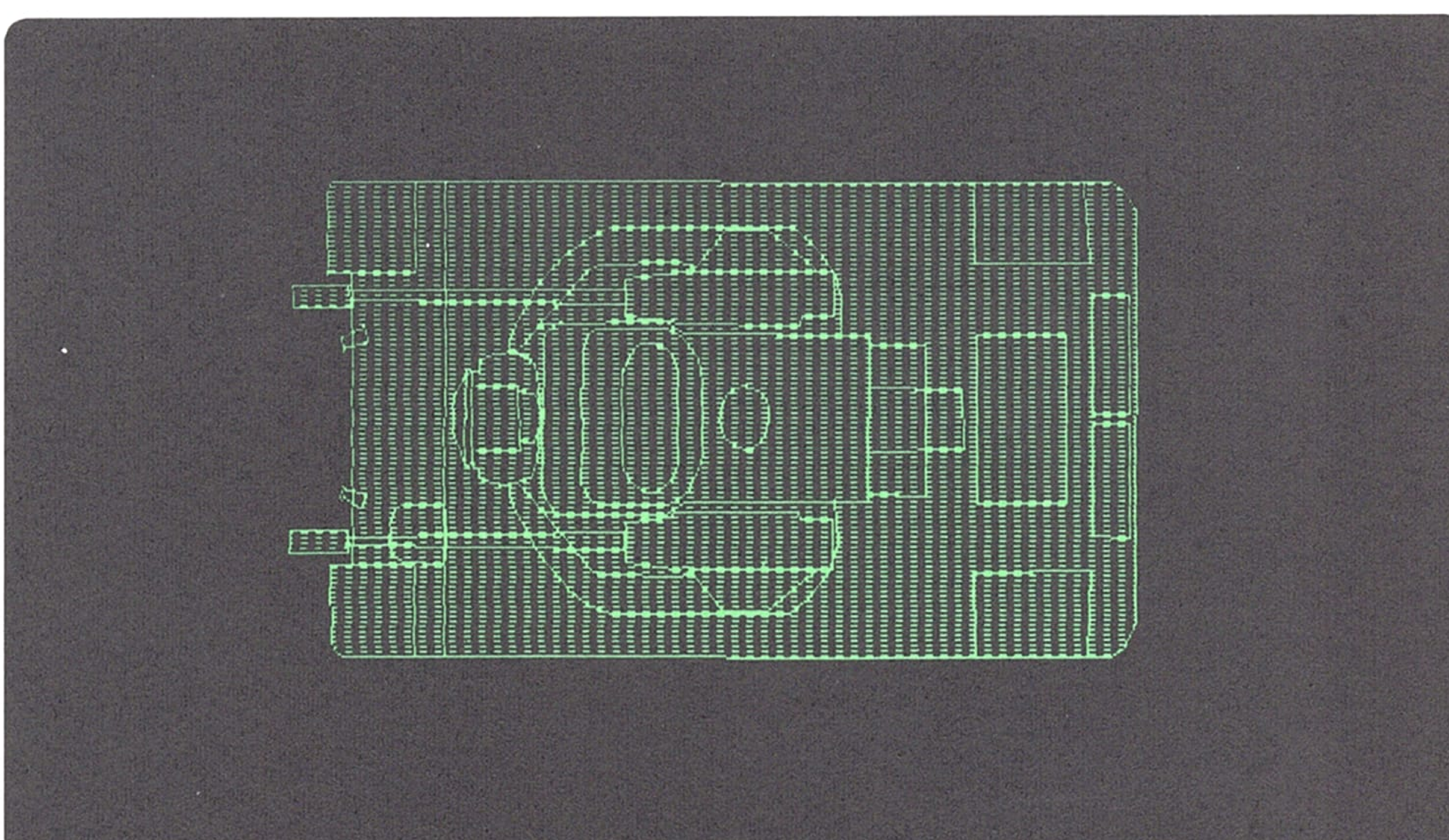


●プログラム 3−12●

```
100 FOR I=1 TO 7:COLOR=(I,I):NEXT I
110 SCREEN 0,3:CLS 3:SCREEN 0,0
120 WIDTH 80,25:CONSOLE 0,25,0,1
130 COLOR 7,0,0,7
140 COL=7
150 WINDOW (0,0)-(639,399):VIEW (0,0)-(639,199)
160 '
170 RESTORE *DAT1:GOSUB *DRAW    ←車体部分を描く
180 RESTORE *DAT2:GOSUB *DRAW    ←上部構造物を描く
190 RESTORE *DAT3:GOSUB *PCL     ←色を塗る
200 END
210 '
220 *DRAW
230 READ X,Y,F
240 IF F=0 THEN RETURN
250 IF PCNT=0 OR F<>1 THEN PSET(X,Y),COL
    :GOTO 270
260 LINE(XOLD,YOLD)-(X,Y),COL
270 XOLD=X:YOLD=Y
280 PCNT=1
290 GOTO 230
300 '
310 *PCL
320 READ X,Y
330 IF X=0 AND Y=0 THEN RETURN
340 PAINT (X,Y),CHR$(AA)+CHR$(0)+CHR$(15)
    +CHR$(FF)+CHR$(0)+CHR$(AA),7
350 GOTO 320
360 '
990 *DAT1
1000 DATA 43,202,2,43,168,1,44,165,1,46,164,1,
     464,165,1,471,168,1,478,176,1
1010 DATA 477,344,1,471,353,1,463,357,1,50,356,
     1,47,354,1,45,351,1,44,319,1
1020 DATA 56,319,1,56,201,1,43,201,1,0,0,0
1990 *DAT2
2000 DATA 144,233,2,155,224,1,161,222,1,224,
     221,1,229,223,1,235,226,1,334,227,1
2010 DATA 333,294,1,236,294,1,234,294,1,230,
     296,1,227,298,1,223,299,1,167,299,1
2020 DATA 161,298,1,156,296,1,145,286,1,0,0,0
4990 *DAT3
5000 DATA 71,185,0,0
```

DATAは簡単のため全部打ち込んでありませんので、この状態での実行結果は次のようになります。皆さんはデータを完全にして下さい。





ここではPAINT TILE$文は一種類しか使っていませんが、場所によって塗りわけると面白いと思います。色を塗るのに必要な全ての座標はDATA 3に入っていますので試みて下さい。

●DATA 1●

```
1000  DATA 43 , 202 , 2
1010  DATA 43 , 168 , 1
1020  DATA 44 , 165 , 1
1030  DATA 46 , 164 , 1
1040  DATA 464 , 165 , 1
1050  DATA 471 , 168 , 1
1060  DATA 478 , 176 , 1
1070  DATA 477 , 344 , 1
1080  DATA 471 , 353 , 1
1090  DATA 463 , 357 , 1
1100  DATA 50 , 356 , 1
1110  DATA 47 , 354 , 1
1120  DATA 45 , 351 , 1
1130  DATA 44 , 319 , 1
1140  DATA 56 , 319 , 1
1150  DATA 56 , 201 , 1
1160  DATA 43 , 201 , 1
1170  DATA 392 , 227 , 2
1180  DATA 392 , 295 , 1
1190  DATA 440 , 295 , 1
1200  DATA 440 , 227 , 1
1210  DATA 392 , 227 , 1
1220  DATA 391 , 357 , 2
1230  DATA 390 , 323 , 1
1240  DATA 453 , 322 , 1
1250  DATA 451 , 357 , 1
1260  DATA 391 , 165 , 2
1270  DATA 391 , 198 , 1
1280  DATA 453 , 198 , 1
1290  DATA 450 , 166 , 1
1300  DATA 106 , 164 , 2
1310  DATA 105 , 356 , 1
1320  DATA 56 , 201 , 2
1330  DATA 90 , 201 , 1
1340  DATA 90 , 164 , 1
1350  DATA 56 , 319 , 2
1360  DATA 89 , 319 , 1
1370  DATA 89 , 355 , 1
1380  DATA 454 , 211 , 2
1390  DATA 454 , 259 , 1
1400  DATA 474 , 259 , 1
1410  DATA 474 , 211 , 1
1420  DATA 454 , 211 , 1
1430  DATA 454 , 263 , 2
1440  DATA 454 , 308 , 1
1450  DATA 474 , 309 , 1
1460  DATA 474 , 263 , 1
1470  DATA 454 , 263 , 1
1480  DATA 75 , 311 , 2
1490  DATA 76 , 302 , 1
1500  DATA 78 , 298 , 1
1510  DATA 82 , 295 , 1
1520  DATA 104 , 295 , 1
1530  DATA 106 , 297 , 1
1540  DATA 106 , 317 , 1
1550  DATA 104 , 319 , 1
1560  DATA 83 , 318 , 1
1570  DATA 77 , 315 , 1
1580  DATA 76 , 306 , 1
1590  DATA 50 , 227 , 2
1600  DATA 51 , 233 , 1
1610  DATA 52 , 231 , 1
1620  DATA 56 , 231 , 1
1630  DATA 64 , 229 , 1
1640  DATA 63 , 223 , 1
1650  DATA 55 , 225 , 1
1660  DATA 52 , 228 , 1
1670  DATA 50 , 227 , 1
1680  DATA 50 , 227 , 1
1690  DATA 51 , 287 , 2
1700  DATA 49 , 292 , 1
1710  DATA 50 , 293 , 1
1720  DATA 51 , 291 , 1
1730  DATA 53 , 293 , 1
1740  DATA 60 , 296 , 1
1750  DATA 63 , 291 , 1
1760  DATA 56 , 289 , 1
1770  DATA 52 , 289 , 1
1780  DATA 52 , 288 , 1
1790  DATA 51 , 287 , 1
```

●DATA 2●

```
2000  DATA 144 , 233 , 2
2010  DATA 155 , 224 , 1
2020  DATA 161 , 222 , 1
2030  DATA 224 , 221 , 1
2040  DATA 229 , 223 , 1
2050  DATA 235 , 226 , 1
2060  DATA 334 , 227 , 1
2070  DATA 333 , 294 , 1
2080  DATA 236 , 294 , 1
2090  DATA 234 , 294 , 1
2100  DATA 230 , 296 , 1
2110  DATA 227 , 298 , 1
2120  DATA 223 , 299 , 1
2130  DATA 167 , 299 , 1
2140  DATA 161 , 298 , 1
2150  DATA 156 , 296 , 1
2160  DATA 145 , 286 , 1
2170  DATA 335 , 231 , 2
2180  DATA 364 , 231 , 1
2190  DATA 364 , 292 , 1
2200  DATA 334 , 291 , 1
2210  DATA 169 , 221 , 2
2220  DATA 164 , 223 , 1
2230  DATA 161 , 225 , 1
2240  DATA 159 , 229 , 1
2250  DATA 159 , 231 , 1
2260  DATA 158 , 288 , 1
2270  DATA 160 , 293 , 1
2280  DATA 163 , 296 , 1
2290  DATA 165 , 298 , 1
2300  DATA 167 , 298 , 1
2310  DATA 254 , 260 , 2
2320  DATA 254 , 257 , 1
2330  DATA 256 , 253 , 1
2340  DATA 257 , 251 , 1
2350  DATA 260 , 249 , 1
2360  DATA 263 , 248 , 1
2370  DATA 267 , 247 , 1
2380  DATA 271 , 248 , 1
2390  DATA 275 , 249 , 1
2400  DATA 278 , 252 , 1
2410  DATA 280 , 256 , 1
2420  DATA 280 , 261 , 1
2430  DATA 279 , 265 , 1
2440  DATA 277 , 268 , 1
2450  DATA 275 , 270 , 1
2460  DATA 272 , 272 , 1
2470  DATA 270 , 273 , 1
2480  DATA 267 , 273 , 1
2490  DATA 263 , 272 , 1
2500  DATA 260 , 270 , 1
2510  DATA 257 , 269 , 1
2520  DATA 255 , 266 , 1
2530  DATA 254 , 263 , 1
2540  DATA 253 , 260 , 1
2550  DATA 179 , 260 , 2
2560  DATA 180 , 234 , 1
2570  DATA 181 , 230 , 1
2580  DATA 183 , 227 , 1
2590  DATA 185 , 225 , 1
2600  DATA 188 , 224 , 1
2610  DATA 190 , 224 , 1
2620  DATA 224 , 224 , 1
2630  DATA 229 , 225 , 1
2640  DATA 233 , 227 , 1
2650  DATA 236 , 230 , 1
2660  DATA 239 , 233 , 1
2670  DATA 242 , 236 , 1
2680  DATA 243 , 241 , 1
2690  DATA 244 , 246 , 1
2700  DATA 244 , 260 , 1
2710  DATA 244 , 275 , 1
2720  DATA 243 , 279 , 1
2730  DATA 240 , 284 , 1
2740  DATA 238 , 288 , 1
2750  DATA 234 , 291 , 1
2760  DATA 229 , 294 , 1
2770  DATA 224 , 295 , 1
2780  DATA 190 , 295 , 1
2790  DATA 185 , 294 , 1
2800  DATA 182 , 291 , 1
2810  DATA 181 , 289 , 1
2820  DATA 179 , 287 , 1
2830  DATA 180 , 260 , 1
2840  DATA 201 , 242 , 2
```

```
2850  DATA 204 , 236 , 1
2860  DATA 207 , 233 , 1
2870  DATA 211 , 230 , 1
2880  DATA 217 , 230 , 1
2890  DATA 223 , 231 , 1
2900  DATA 226 , 233 , 1
2910  DATA 229 , 237 , 1
2920  DATA 231 , 241 , 1
2930  DATA 231 , 278 , 1
2940  DATA 229 , 284 , 1
2950  DATA 223 , 288 , 1
2960  DATA 218 , 290 , 1
2970  DATA 215 , 290 , 1
2980  DATA 211 , 289 , 1
2990  DATA 208 , 287 , 1
3000  DATA 205 , 285 , 1
3010  DATA 203 , 282 , 1
3020  DATA 201 , 279 , 1
3030  DATA 201 , 243 , 1
3040  DATA 201 , 240 , 1
3050  DATA 144 , 234 , 2
3060  DATA 152 , 237 , 1
3070  DATA 154 , 238 , 1
3080  DATA 157 , 253 , 1
3090  DATA 157 , 260 , 1
3100  DATA 156 , 271 , 1
3110  DATA 154 , 281 , 1
3120  DATA 154 , 281 , 1
3130  DATA 153 , 283 , 1
3140  DATA 151 , 284 , 1
3150  DATA 144 , 286 , 1
3160  DATA 138 , 287 , 1
3170  DATA 131 , 286 , 1
3180  DATA 124 , 284 , 1
3190  DATA 124 , 280 , 1
3200  DATA 121 , 280 , 1
3210  DATA 121 , 278 , 1
3220  DATA 116 , 279 , 1
3230  DATA 114 , 274 , 1
3240  DATA 112 , 269 , 1
3250  DATA 110 , 265 , 1
3260  DATA 110 , 260 , 1
3270  DATA 110 , 256 , 1
3280  DATA 111 , 251 , 1
3290  DATA 113 , 247 , 1
3300  DATA 117 , 242 , 1
3310  DATA 120 , 241 , 1
3320  DATA 124 , 241 , 1
3330  DATA 124 , 237 , 1
3340  DATA 129 , 235 , 1
3350  DATA 132 , 234 , 1
3360  DATA 136 , 234 , 1
3370  DATA 142 , 233 , 1
3380  DATA 146 , 234 , 1
3390  DATA 152 , 236 , 1
3400  DATA 120 , 241 , 2
3410  DATA 120 , 241 , 2
3420  DATA 120 , 279 , 1
3430  DATA 116 , 279 , 2
3440  DATA 117 , 241 , 1
3450  DATA 122 , 273 , 2
3460  DATA 146 , 273 , 1
3470  DATA 145 , 247 , 1
3480  DATA 122 , 247 , 1
3490  DATA 145 , 249 , 2
3500  DATA 154 , 250 , 1
3510  DATA 157 , 254 , 1
3520  DATA 158 , 260 , 1
3530  DATA 156 , 266 , 1
3540  DATA 154 , 270 , 1
3550  DATA 146 , 269 , 1
3560  DATA 143 , 234 , 2
3570  DATA 182 , 201 , 1
3580  DATA 187 , 199 , 1
3590  DATA 188 , 198 , 1
3600  DATA 231 , 197 , 1
3610  DATA 240 , 199 , 1
3620  DATA 145 , 286 , 2
3630  DATA 182 , 319 , 1
3640  DATA 185 , 322 , 1
3650  DATA 191 , 323 , 1
3660  DATA 232 , 324 , 1
3670  DATA 239 , 322 , 1
3680  DATA 140 , 233 , 2
3690  DATA 140 , 228 , 1
3700  DATA 154 , 208 , 1
```

```
3710  DATA 169 , 195 , 1
3720  DATA 184 , 187 , 1
3730  DATA 192 , 184 , 1
3740  DATA 294 , 183 , 2
3750  DATA 194 , 183 , 2
3760  DATA 294 , 183 , 1
3770  DATA 304 , 190 , 1
3780  DATA 315 , 201 , 1
3790  DATA 319 , 211 , 1
3800  DATA 319 , 226 , 1
3810  DATA 140 , 287 , 2
3820  DATA 140 , 292 , 1
3830  DATA 154 , 312 , 1
3840  DATA 163 , 321 , 1
3850  DATA 172 , 329 , 1
3860  DATA 182 , 335 , 1
3870  DATA 191 , 338 , 1
3880  DATA 196 , 339 , 1
3890  DATA 291 , 339 , 1
3900  DATA 301 , 333 , 1
3910  DATA 312 , 322 , 1
3920  DATA 317 , 310 , 1
3930  DATA 315 , 293 , 1
3940  DATA 334 , 248 , 2
3950  DATA 384 , 249 , 1
3960  DATA 385 , 273 , 1
3970  DATA 333 , 274 , 1
3980  DATA 24 , 215 , 2
3990  DATA 24 , 206 , 1
4000  DATA 54 , 206 , 1
4010  DATA 55 , 215 , 1
4020  DATA 24 , 215 , 1
4030  DATA 23 , 305 , 2
4040  DATA 53 , 305 , 1
4050  DATA 54 , 314 , 1
4060  DATA 22 , 314 , 1
4070  DATA 22 , 305 , 1
4080  DATA 54 , 208 , 2
4090  DATA 203 , 208 , 1
4100  DATA 203 , 214 , 1
4110  DATA 54 , 212 , 1
4120  DATA 53 , 308 , 2
4130  DATA 202 , 306 , 1
4140  DATA 202 , 313 , 1
4150  DATA 52 , 311 , 1
4160  DATA 203 , 219 , 2
4170  DATA 203 , 204 , 1
4180  DATA 232 , 201 , 1
4190  DATA 239 , 201 , 1
4200  DATA 257 , 184 , 1
4210  DATA 280 , 184 , 1
4220  DATA 304 , 199 , 1
4230  DATA 306 , 201 , 1
4240  DATA 315 , 201 , 1
4250  DATA 315 , 219 , 1
4260  DATA 298 , 219 , 1
4270  DATA 297 , 223 , 1
4280  DATA 230 , 222 , 1
4290  DATA 229 , 221 , 1
4300  DATA 203 , 219 , 1
4310  DATA 203 , 302 , 2
4320  DATA 203 , 318 , 1
4330  DATA 230 , 321 , 1
4340  DATA 236 , 320 , 1
4350  DATA 255 , 338 , 1
4360  DATA 279 , 338 , 1
4370  DATA 306 , 321 , 1
4380  DATA 314 , 321 , 1
4390  DATA 314 , 301 , 1
4400  DATA 297 , 301 , 1
4410  DATA 296 , 298 , 1
4420  DATA 229 , 298 , 1
4430  DATA 228 , 300 , 1
4440  DATA 204 , 302 , 1
4450  DATA 202 , 302 , 1
4460  DATA 239 , 201 , 2
4470  DATA 308 , 201 , 1
4480  DATA 237 , 321 , 2
4490  DATA 307 , 321 , 1
```

●DATA 3●

```
5000    DATA 71 , 185
5010    DATA 85 , 250
5020    DATA 68 , 338
5030    DATA 96 , 334
5040    DATA 100 , 245
5050    DATA 91 , 303
5060    DATA 129 , 329
5070    DATA 419 , 337
5080    DATA 419 , 255
5090    DATA 413 , 184
5100    DATA 466 , 238
5110    DATA 465 , 284
5120    DATA 376 , 262
5130    DATA 345 , 262
5140    DATA 350 , 239
5150    DATA 342 , 283
5160    DATA 299 , 253
5170    DATA 264 , 260
5180    DATA 215 , 253
5190    DATA 169 , 257
5200    DATA 190 , 275
5210    DATA 233 , 311
5220    DATA 251 , 212
5230    DATA 267 , 193
5240    DATA 266 , 327
5250    DATA 198 , 191
5260    DATA 175 , 322
5270    DATA 193 , 318
5280    DATA 174 , 301
5290    DATA 169 , 218
5300    DATA 188 , 203
5310    DATA 106 , 211
5320    DATA 161 , 210
5330    DATA 189 , 210
5340    DATA 187 , 309
5350    DATA 162 , 309
5360    DATA 113 , 309
5370    DATA 87 , 211
5380    DATA 87 , 309
5390    DATA 41 , 311
5400    DATA 39 , 211
5410    DATA 96 , 202
5420    DATA 130 , 211
5430    DATA 125 , 222
5440    DATA 67 , 315
5450    DATA 87 , 315
5460    DATA 114 , 256
5470    DATA 132 , 254
5480    DATA 136 , 239
5490    DATA 153 , 260
5500    DATA 142 , 280
5510    DATA 150 , 296
5520    DATA 155 , 288
5530    DATA 155 , 230
5540    DATA 153 , 222
5550    DATA 294 , 189
5560    DATA 301 , 223
5570    DATA 302 , 298
5580    DATA 296 , 331
5590    DATA 59 , 310
```

3·5 タイル文によらない色の作り方

PAINT TILE$文によらなければ、中間色を合成することはできないでしょうか？いろいろなやり方が考えられますが、CIRCLE文で円を作り、それをPAINTする方法もあります。カラー印刷のやり方を真似たものです。青、赤、緑の３原色をわずかにずらして重ねるのです。色によっては青、赤、緑の円の大きさが異なることになります。たとえば赤なら、ほとんどすべての領域が赤の円でうめ尽され、青や緑の円はわずかにとどまるか、まったくなくなってしまうかのどちらかですし、橙を作るには黄色と赤の円を大きくすることになります。

このやり方でも中間色を作ることができます。プログラムを作ってみました。R1、R2、R3はそれぞれ青、赤、緑の円の半径です。これをいろいろ変えてやると、さまざまな色を楽しむことができます。

●プログラム 3—13●

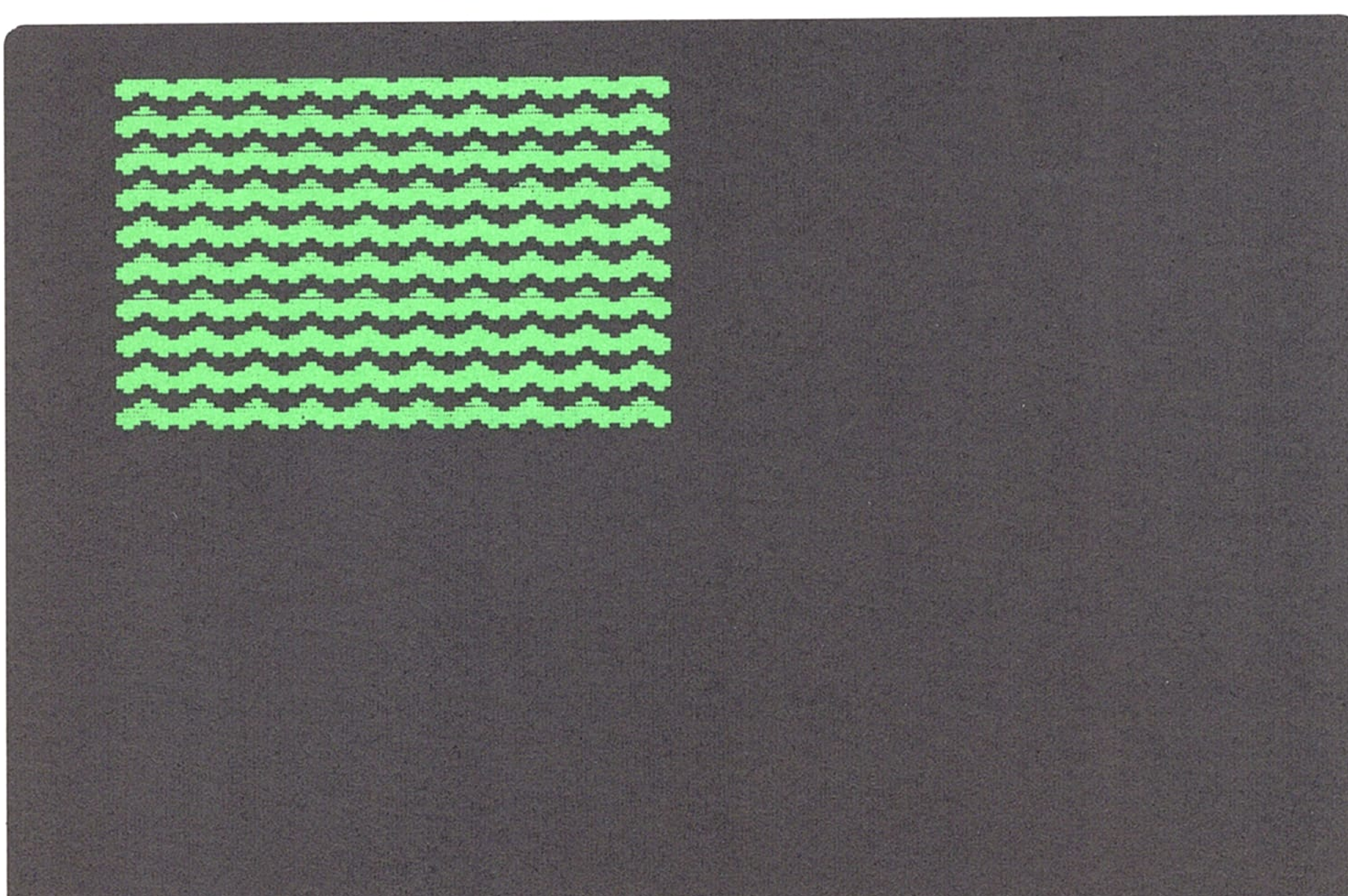
```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WIDTH 80,25:CONSOLE 0,25,0,1
120 '
130 INPUT "R1,R2,R3 ",R1,R2,R3
140 CLS 3
150 FOR I=1 TO 10                     ┐
160 FOR J=1 TO 10                     │
170 CIRCLE(16*I,8*J),R1,1             ├ 100個円を描く
180 NEXT J                            │
190 NEXT I                            ┘
200 '
210 FOR I=1 TO 10                     ┐
220 FOR J=1 TO 10                     │
230 PAINT(16*I,8*J),1,1               ├ 円の内部を青く塗る
240 NEXT J                            │
250 NEXT I                            ┘
260 '
270 FOR I=1 TO 10                     ┐
280 FOR J=1 TO 10                     │
290 CIRCLE(16*I-5,8*J+2),R2,2         ├ 中心を少しずらして100個の円を描く
300 NEXT J                            │
310 NEXT I                            ┘
320 '
```

```
330 FOR I=1 TO 10
340 FOR J=1 TO 10
350 PAINT(16*I-5,8*J+2),2,2
360 NEXT J
370 NEXT I
380 ′
390 FOR I=1 TO 10
400 FOR J=1 TO 10
410 CIRCLE(16*I+5,8*J+2),R3,4
420 NEXT J
430 NEXT I
440 ′
450 FOR I=1 TO 10
460 FOR J=1 TO 10
470 PAINT(16*I+5,8*J+2),4,4
480 NEXT J
490 NEXT I
500 ′
510 GOTO 130
```

（330〜370）いま描いた100個の円を赤で塗る
青と赤の重なったところは黄色になる

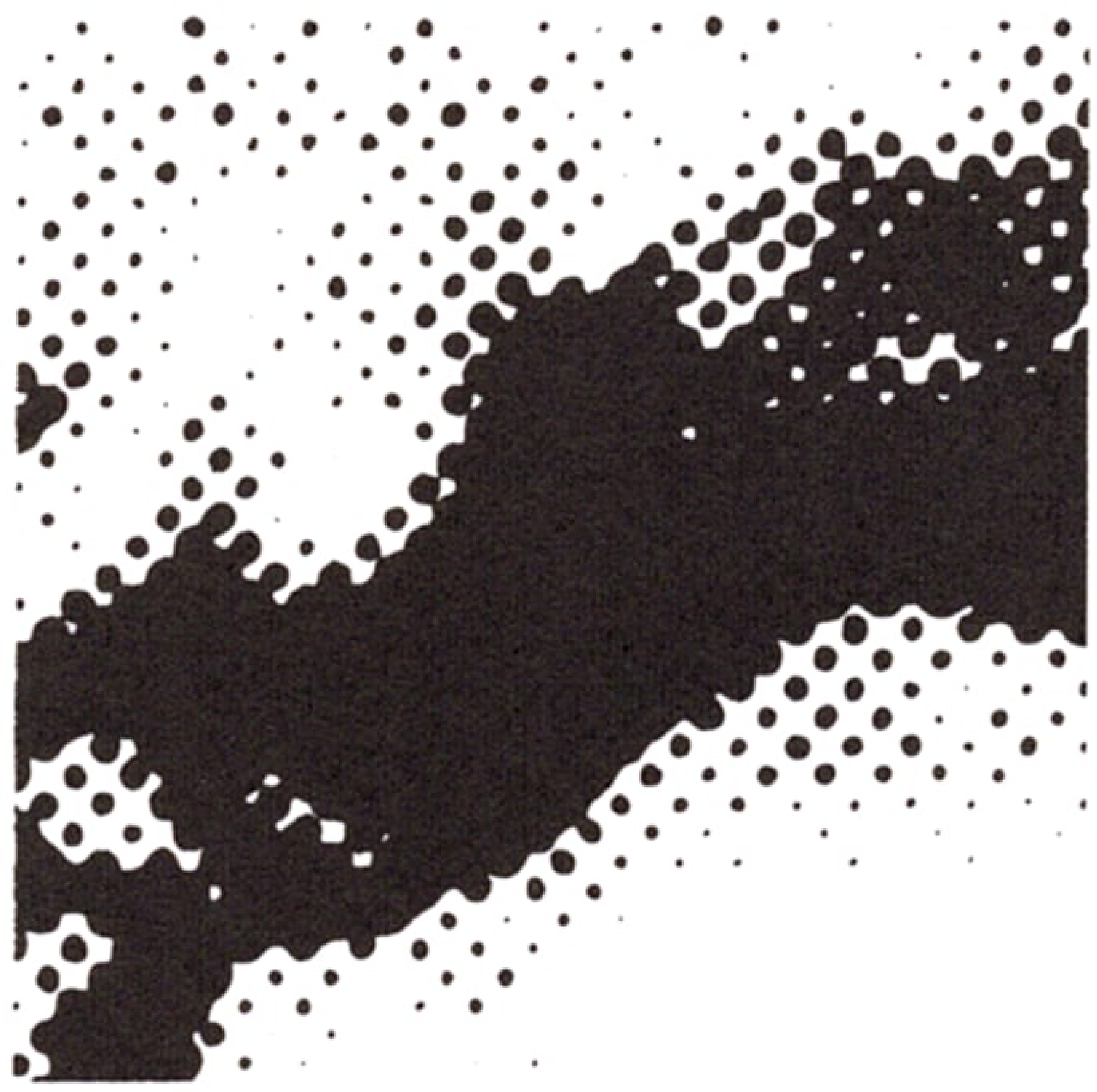
（390〜490）さらに中心をずらして100個円を描く
3原色の組み合わせによりいろいろな色が表示される

実行結果を次に示します。
みなさんも3つの円の半径をいろいろ変えて試してみてください。



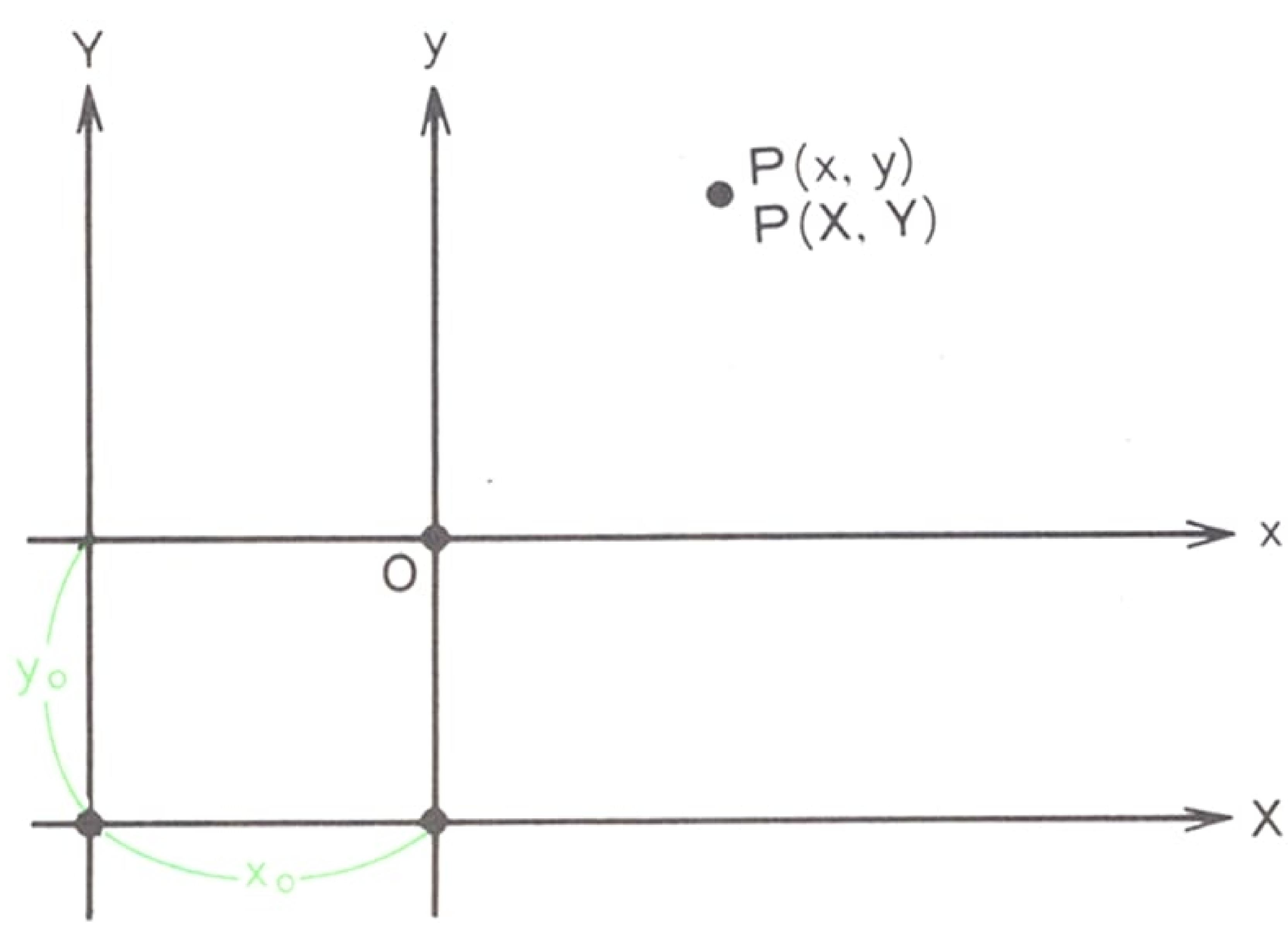
第4章

2次元グラフィックスの世界



4·0 平行移動とその応用

2次元グラフィックスにおいて重要なのは平行移動・回転・拡大縮小の3つです。この3つがあれば基本的には2次元平面でのグラフィックスは実行することができます。そこでまず平行移動から勉強することにしましょう。



同一の点を左図のような2つの座標系xy、XYでみた時に、それぞれ、

(x, y)

(X, Y)

であったとします(旧座標系をxy、新座標系をXYとすることを暗黙に仮定して話を進めます)。

本来は座標軸xy、XYを混ぜるのは混乱のもとなので良くないのですが、簡単のため、このように記すことにします。

すると(x, y)と(X, Y)の間には次のような関係があります。

$$\begin{cases} X = x + x_0 \\ Y = y + y_0 \end{cases} \quad \cdots\cdots\cdots\cdots\cdots(1)$$

これは中学校や高校で習う有名なものです。読者の皆さんの中にはアアまた出て来たかとガッカリされる方もあるでしょう。しかし受験数学ではあれほど無味乾燥に見えたこの式が大変役に立つのです。実際に計算するには(1)を使うのですが、もう少し美しく表現するには、行列というものを使うと便利です。これを使うと(1)は次のように表現できます。

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \cdots\cdots\cdots\cdots\cdots(2)$$

今はこれらの中味に深く立ち入ることはしません。(1)式は次のように読むことができます。(x, y)にある点を(X, Y)に動かすにはx_0とy_0を加えれば良いのです。そこで旧座標 xy と新座標系の間の平行移動の関係がはっきりしました。

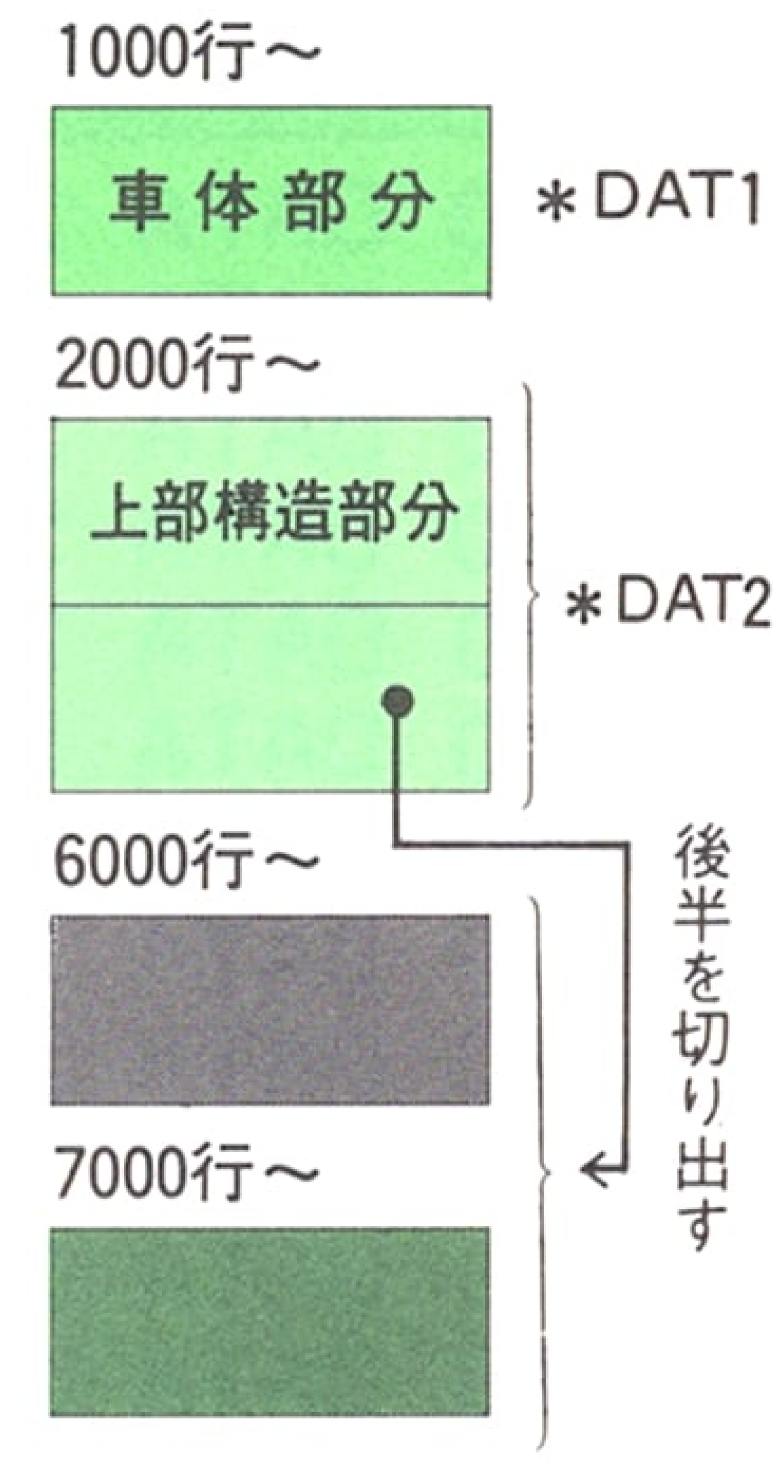
これを使っておもしろい実験をしてみます。ある図形を x_0 だけ移

動するには(1)に従って、その図形の構成要素の各々に x_0 を加えてやればよいわけですから、4－1のようなプログラムを組んでみました。

ここで行番号6000からのデータは3・4節の上部構造部分のデータ(＊DAT 2)一部を組み変えて切り離してあります。少々面倒な仕事ですが、挑戦してみて下さい。

このようなことをするのは＊DAT 2 を作る時に私の失敗があり、上側可動部と下側可動部が明確に分離していなかったからです。

平行移動したデータは3つの画面に分けて書き込まれます。



第1画面　SCREEN 1,0,0,1　――下側可動部分前方へ
第2画面　SCREEN 1,0,1,2　――両側可動部分並ぶ
第3画面　SCREEN 1,0,2,4　――上側可動部分前方へ

●プログラム 4－1●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 '
120 WINDOW (0,0)-(639,399):VIEW (0,0)-(639,199)
130 WIDTH 80,25:CONSOLE 0,25,0,1
140 COLOR 1,0,0,7:COL=7
150 '
160 SCREEN 1,0,0,1:RESTORE 6000:X0= 20:GOSUB *GDRAW
170 SCREEN 1,0,0,1:RESTORE 7000:X0=-20:GOSUB *GDRAW
180 SCREEN 1,0,1,2:RESTORE 6000:X0=  0:GOSUB *GDRAW
190 SCREEN 1,0,1,2:RESTORE 7000:X0=  0:GOSUB *GDRAW
200 SCREEN 1,0,2,4:RESTORE 6000:X0=-20:GOSUB *GDRAW
210 SCREEN 1,0,2,4:RESTORE 7000:X0= 20:GOSUB *GDRAW
220 '
230 FOR K=1 TO 1000
240 SCREEN 1,,,1:FOR I=1 TO 100:NEXT I
250 SCREEN 1,,,2:FOR I=1 TO 100:NEXT I
260 SCREEN 1,,,4:FOR I=1 TO 100:NEXT I
270 NEXT K
280 END
290 '
300 *GDRAW
310 READ X,Y,F
320 IF X=0 AND Y=0 THEN RETURN
330 X=INT(X/6.72+.5):Y=199-INT(Y/7!+.5)
340 IF F>1 THEN PSET(X+X0,Y),COL:GOTO 360
350 LINE(XOLD+X0,YOLD)-(X+X0,Y),COL
360 XOLD=X:YOLD=Y
370 GOTO 310
380 '
```

●DATA文は次ページに示します。続けて打ち込んでください。

●DATA文●

```
6000 DATA 24 , 215 , 2
6010 DATA 24 , 206 , 1
6020 DATA 54 , 206 , 1
6030 DATA 55 , 215 , 1
6040 DATA 24 , 215 , 1
6050 DATA 54 , 208 , 2
6060 DATA 203 , 208 , 1
6070 DATA 203 , 214 , 1
6080 DATA 54 , 212 , 1
6090 DATA 203 , 219 , 2
6100 DATA 203 , 204 , 1
6110 DATA 232 , 201 , 1
6120 DATA 239 , 201 , 1
6130 DATA 257 , 184 , 1
6140 DATA 280 , 184 , 1
6150 DATA 304 , 199 , 1
6160 DATA 306 , 201 , 1
6170 DATA 315 , 201 , 1
6180 DATA 315 , 219 , 1
6190 DATA 298 , 219 , 1
6200 DATA 297 , 223 , 1
6210 DATA 230 , 222 , 1
6220 DATA 229 , 221 , 1
6230 DATA 203 , 219 , 1
6240 DATA 239 , 201 , 2
6250 DATA 308 , 201 , 1
6260 DATA 0,0,0
```

```
7000 DATA 23 , 305 , 2
7010 DATA 53 , 305 , 1
7020 DATA 54 , 314 , 1
7030 DATA 22 , 314 , 1
7040 DATA 22 , 305 , 1
7050 DATA 53 , 308 , 2
7060 DATA 202 , 306 , 1
7070 DATA 202 , 313 , 1
7080 DATA 52 , 311 , 1
7090 DATA 203 , 302 , 2
7100 DATA 203 , 318 , 1
7110 DATA 230 , 321 , 1
7120 DATA 236 , 320 , 1
7130 DATA 255 , 338 , 1
7140 DATA 279 , 338 , 1
7150 DATA 306 , 321 , 1
7160 DATA 314 , 321 , 1
7170 DATA 314 , 301 , 1
7180 DATA 297 , 301 , 1
7190 DATA 296 , 298 , 1
7200 DATA 229 , 298 , 1
7210 DATA 228 , 300 , 1
7220 DATA 204 , 302 , 1
7230 DATA 202 , 302 , 1
7240 DATA 237 , 321 , 2
7250 DATA 307 , 321 , 1
7260 DATA 0,0,0
```

プログラムが誤りなく入力できれば、プログラムをRUNさせると3つの画面が切り替って表示されます。すると上下の可動部分がピストン運動をすることになります。ある程度なめらかにピストン運動をしているように見えるようにタイマールーチンを工夫してあります。

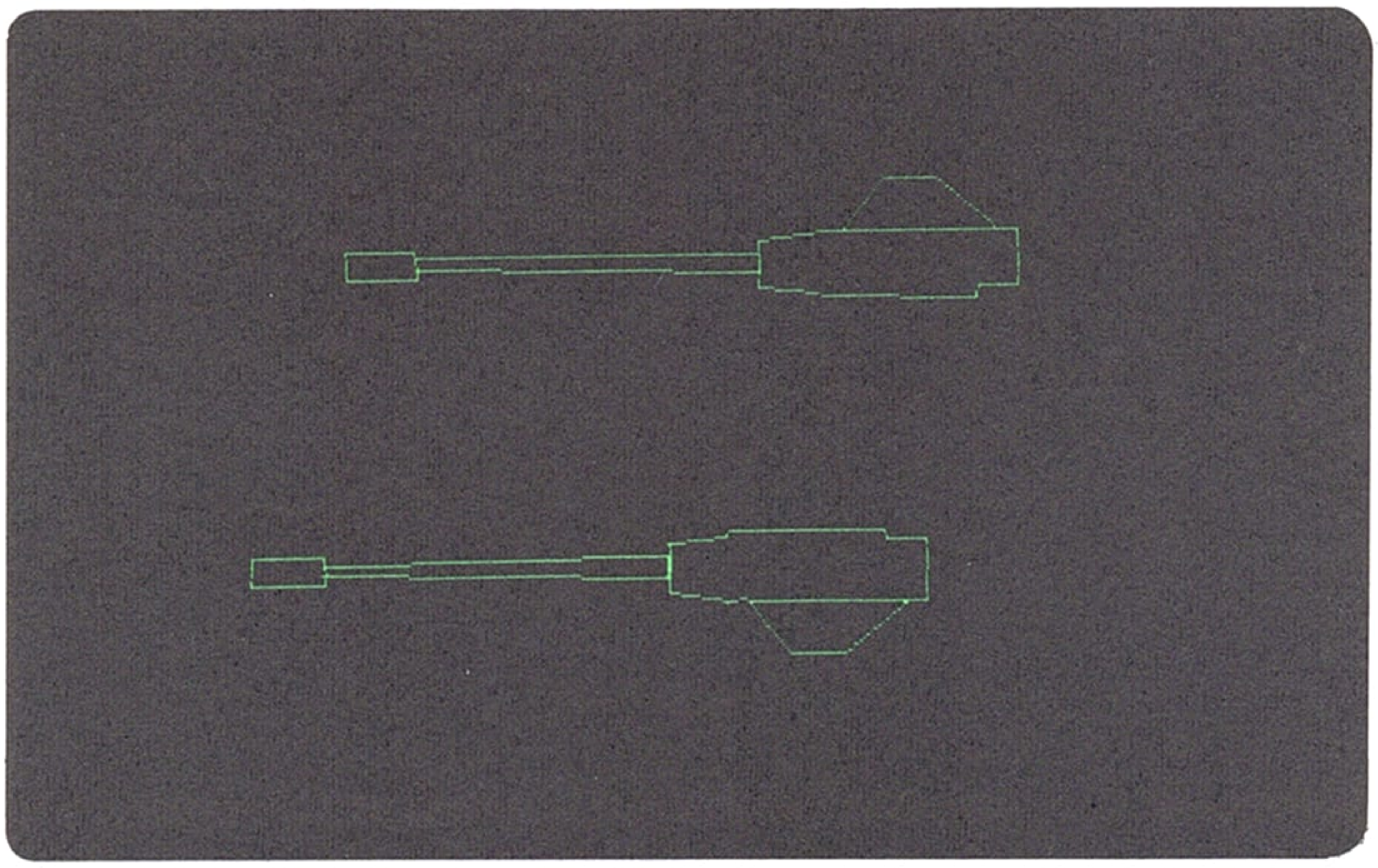
これは上部構造物の可動部分が動きながら排出動作をくり返していることをシュミレートしたのです。

☞ 7 (P.220)

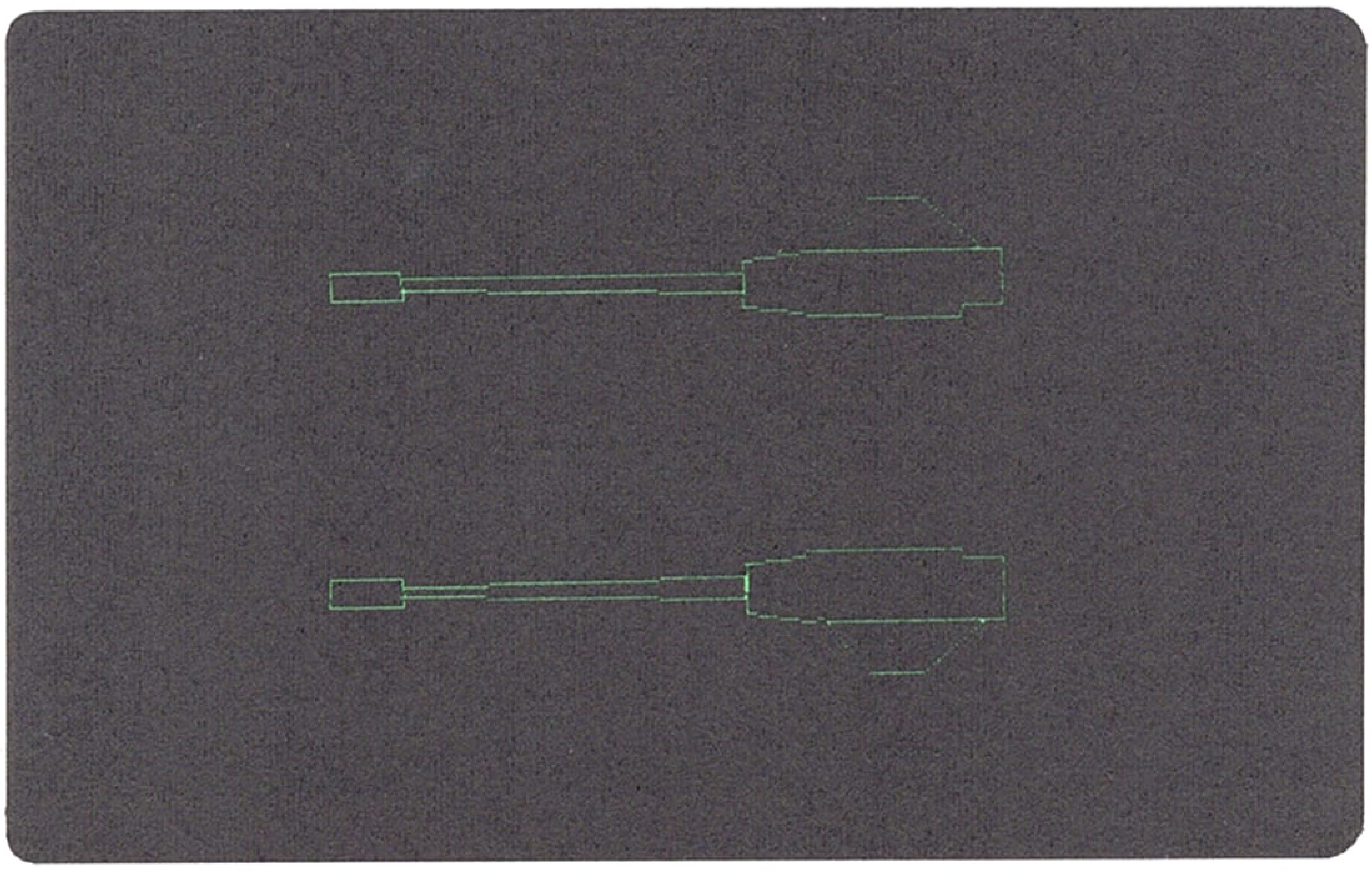
前章ででてきたプログラムを少し改造し、＊DAT 2の部分を削ったものを走らせた後で、このプログラムを走らせますと、138ページに示すようになります。CRT画面上でないと、このプログラムの動きの面白さはわかりにくいので残念です。

車体そのものを動かしても良いのです。この場合は大変簡単ですので読者の皆さんが自分で試してみて下さい。

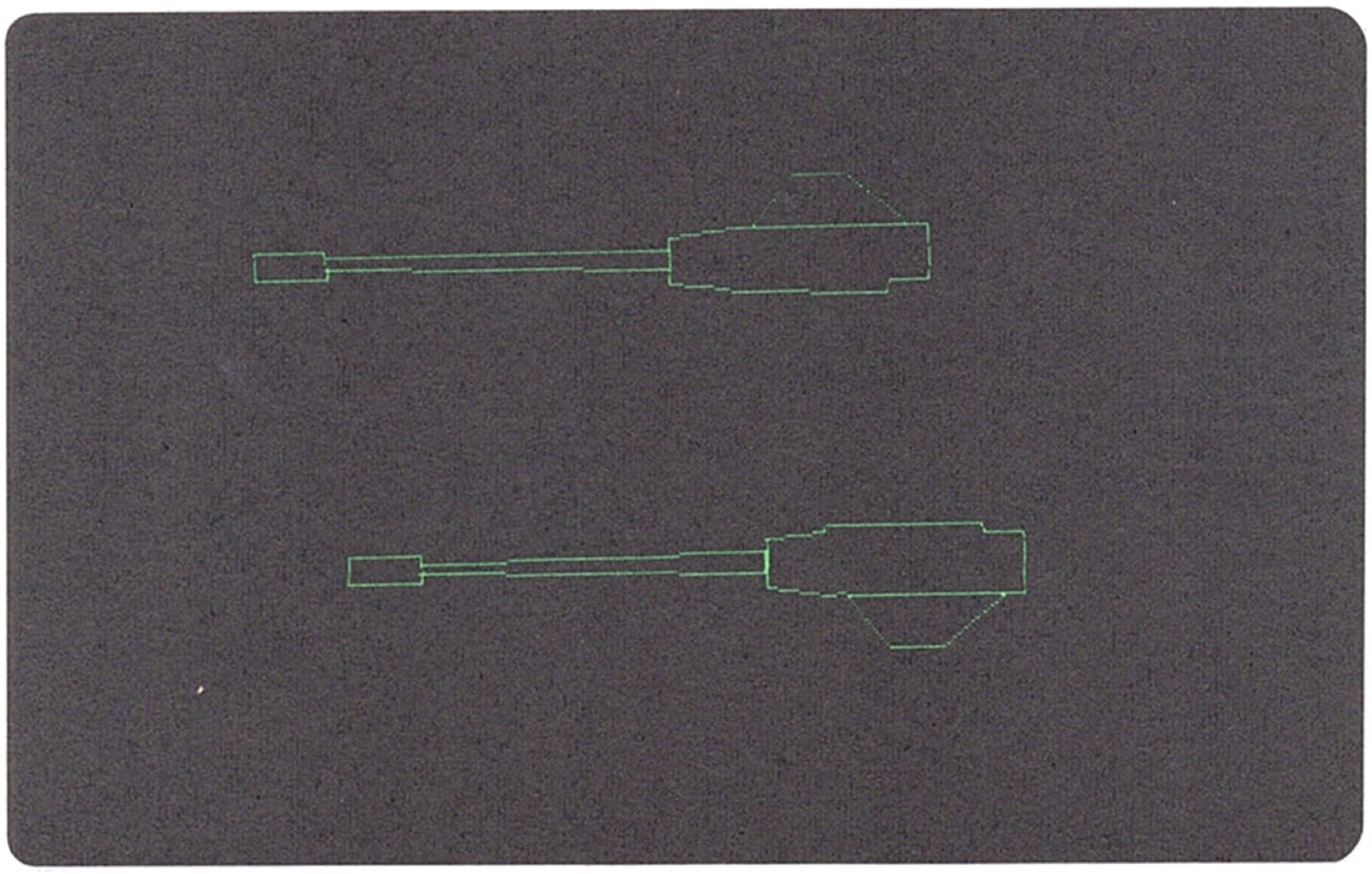
●プログラム 4−1の実行結果



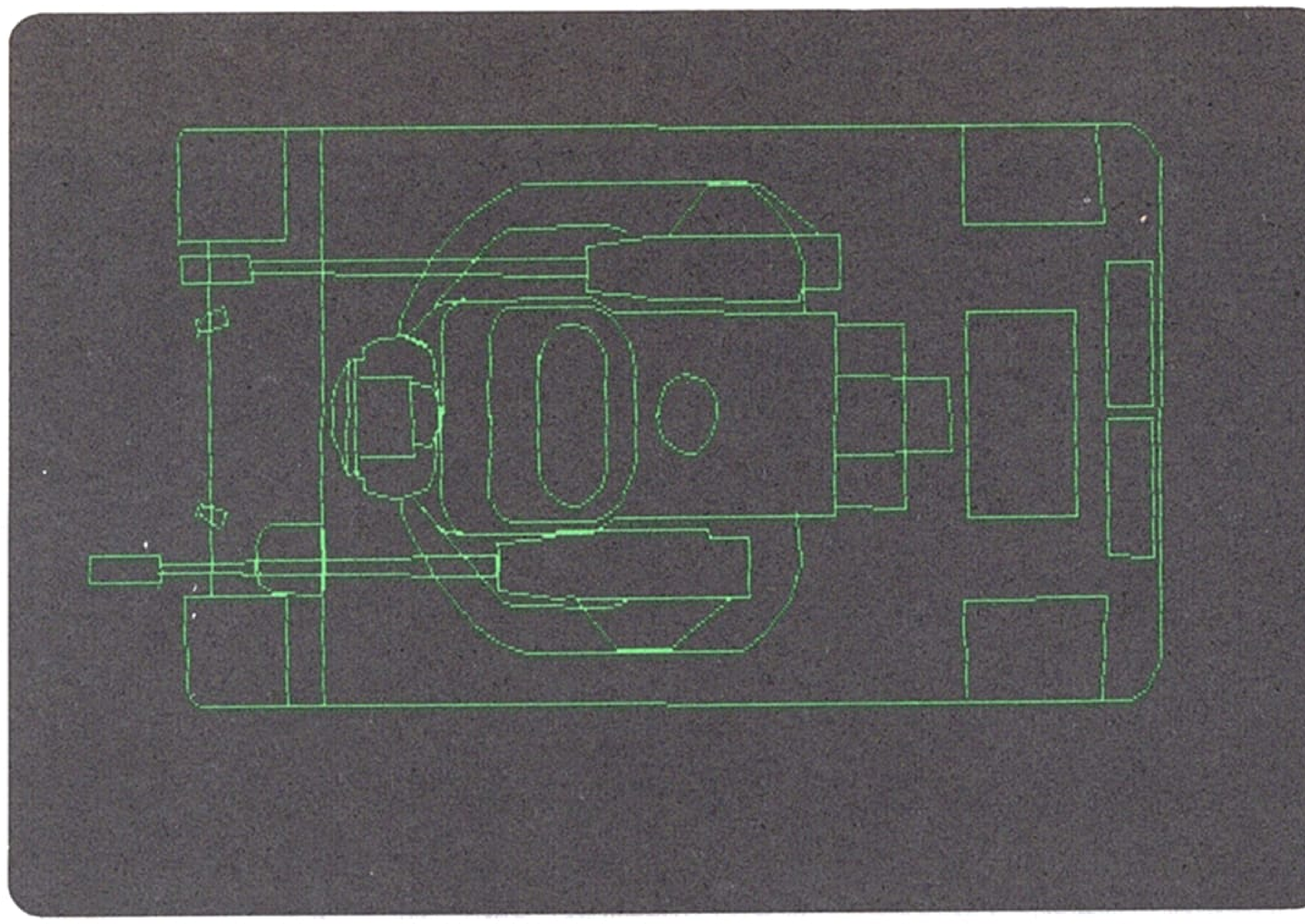
SCREEN 1,,,1



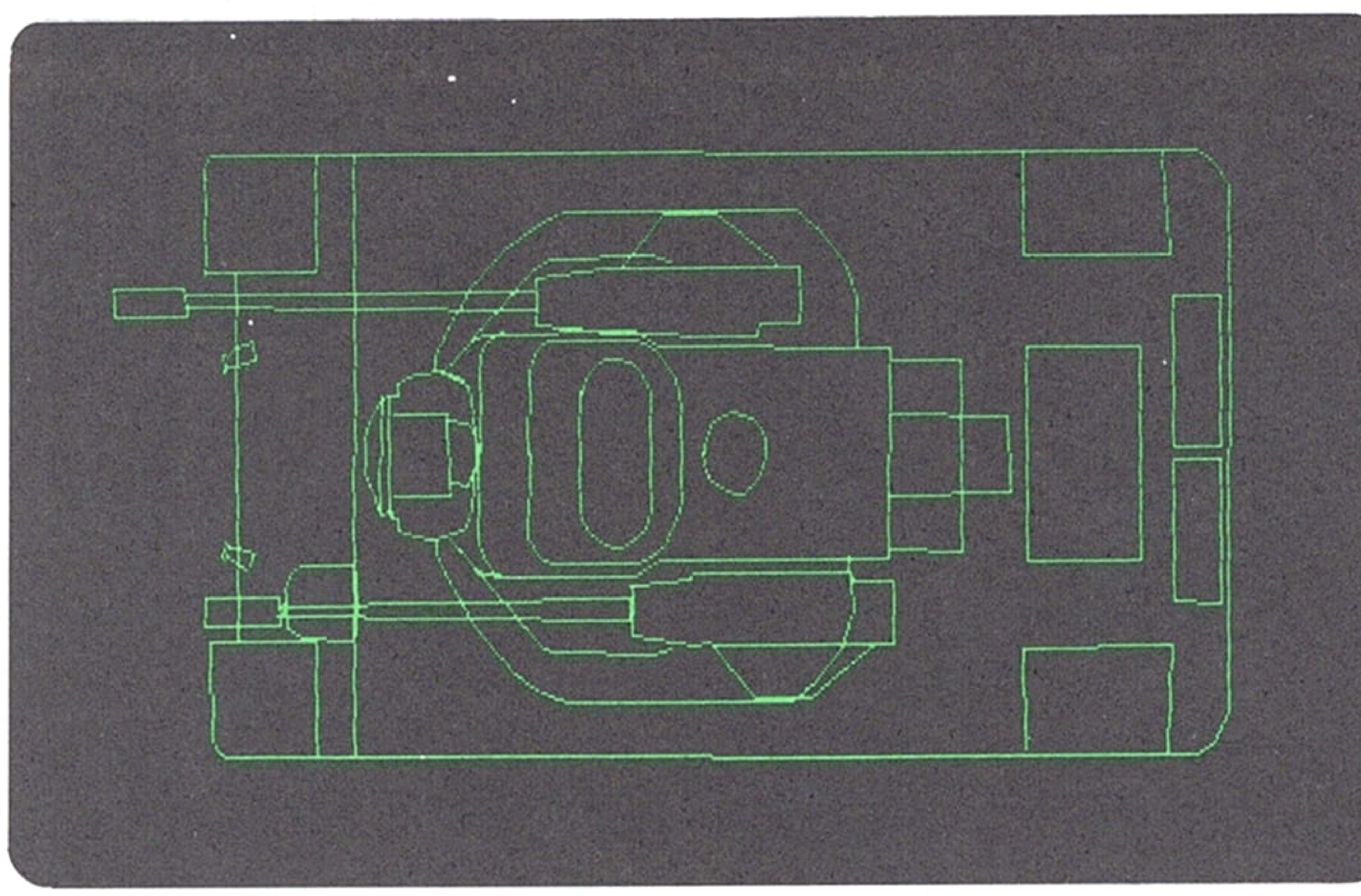
SCREEN 1,,,2



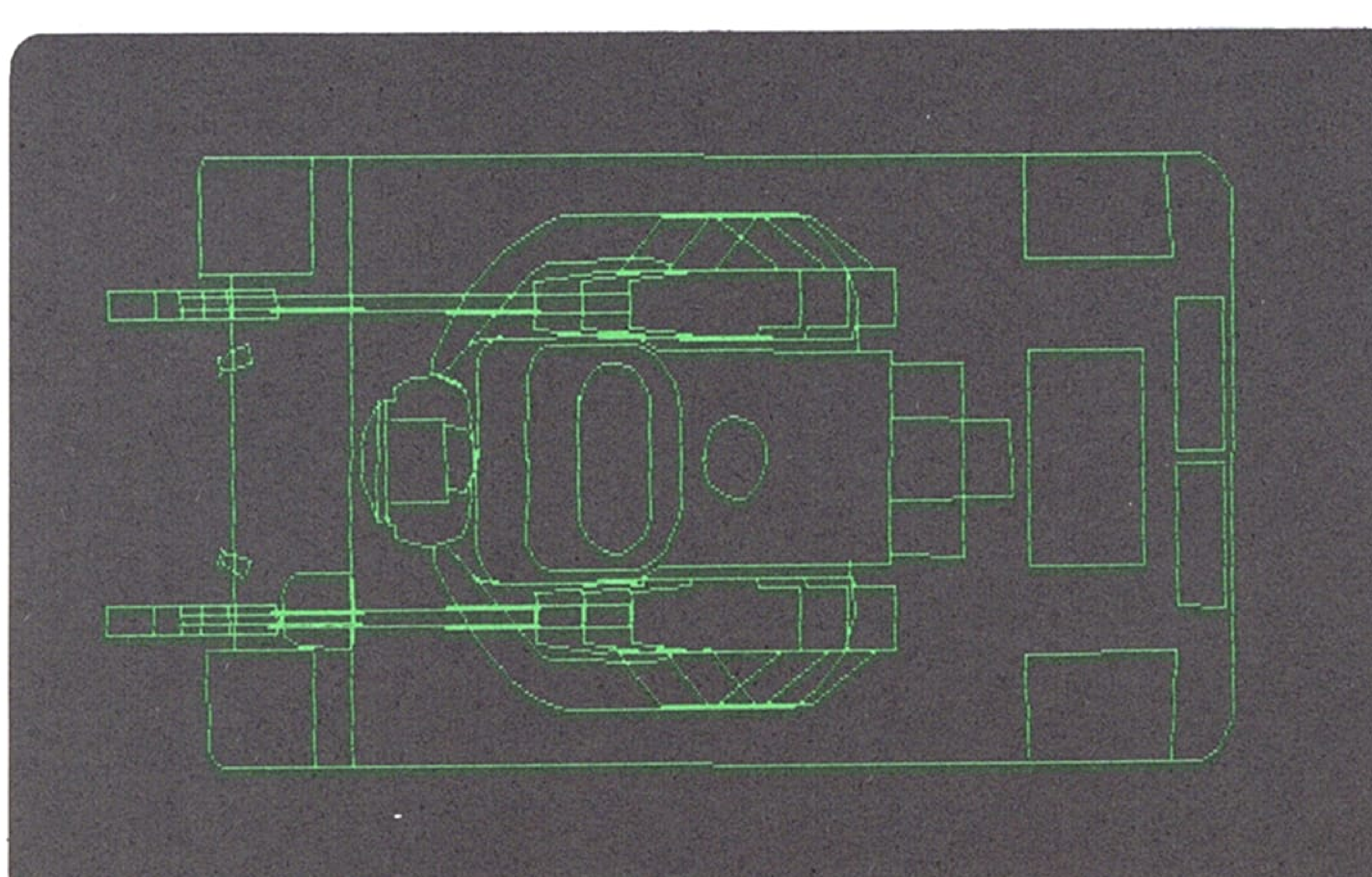
SCREEN 1,,,4



SCREEN 1,,,1



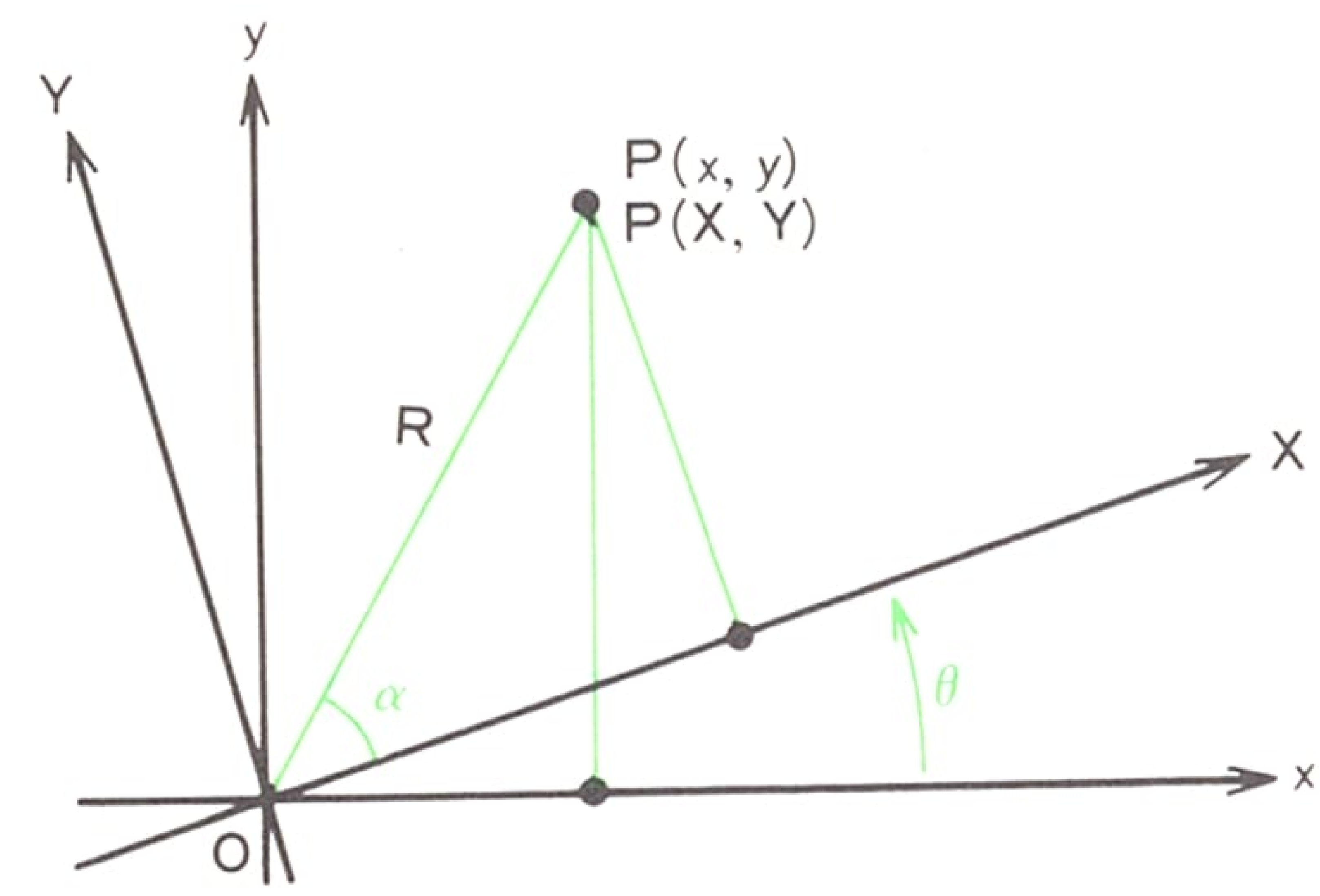
SCREEN 1,,,4



SCREEN 1,,,7

4・1　物体を回転させてみよう

大学受験の数学で受験生が必ず苦しむのは、2次元図形の回転です。ふつうは2次曲線を標準形に変換するものが多く、楕円を回転させるものが多いようです。下の図を見るだけで『あれか！もう駄目だ』などと気弱にならないで下さい。使うことに徹すればそうは難しくはないですし、便利なのです。



ある直角座標系 xy を回転させて新しい座標系XYを作ったとします。この時、2つの座標系の間にどういう関係があるかといいますと次のようになるのでした。

$$X = x\cos\theta + y\sin\theta$$
$$Y = -x\sin\theta + y\cos\theta$$

証明はどうしたか覚えていますか。こういうのは案外面倒でしたね。

$$x = R\cos(\alpha+\theta) = R\cos\alpha\cos\theta - R\sin\alpha\sin\theta$$
$$y = R\sin(\alpha+\theta) = R\sin\alpha\cos\theta + R\cos\alpha\sin\theta$$

ここで、$X = R\cos\alpha$、$Y = R\sin\alpha$ ですから、上式を書き直すと、

$$x = X\cos\theta - Y\sin\theta$$
$$y = X\sin\theta + Y\cos\theta$$

となります。これをX、Yについて解くと最初の式が得られるのでした。

数学ではいつも難しい表現法を教わったものですが、慣れてしまえば簡単です。

行列で書くと次のようになります。

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

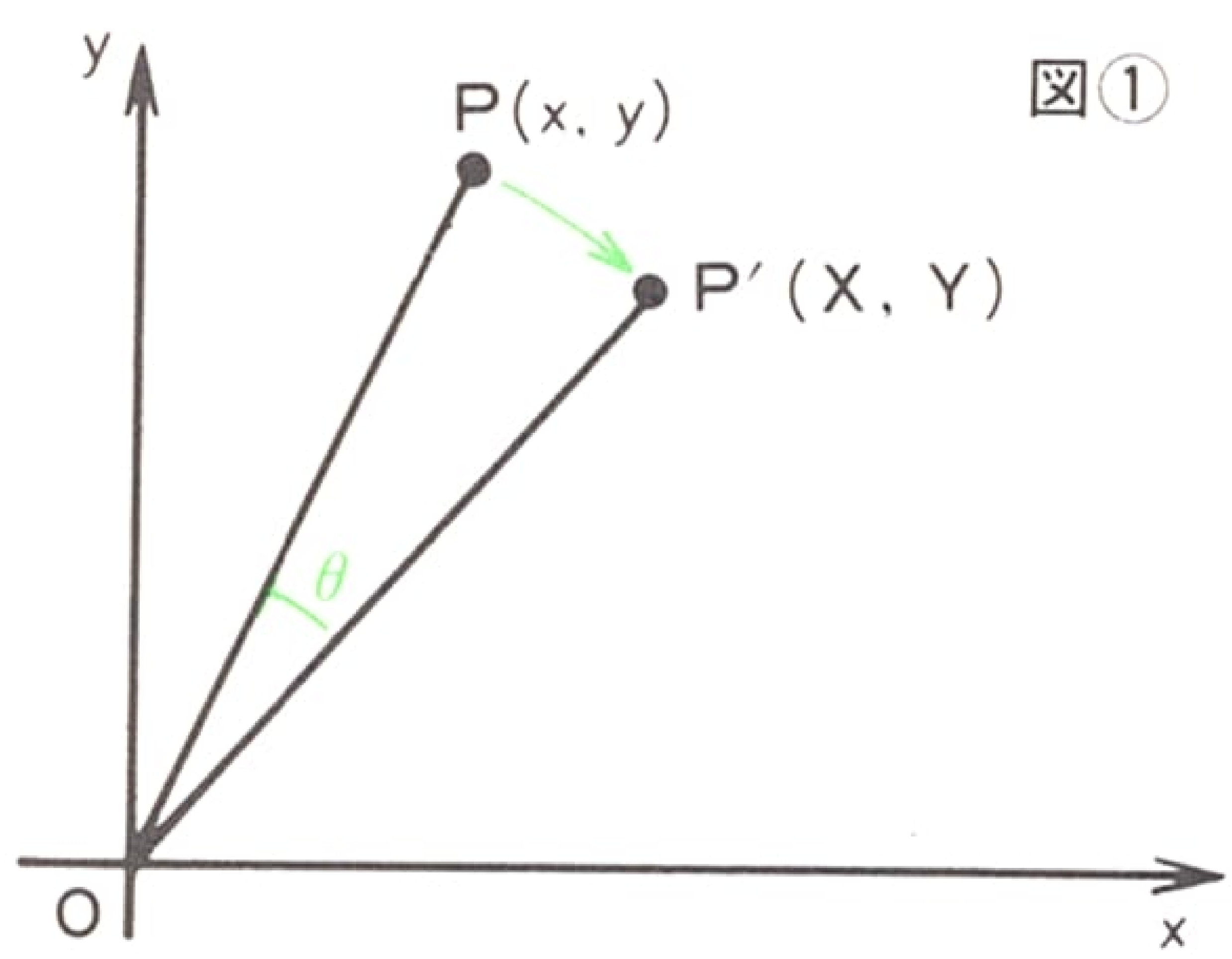
しかし、これを拡張して次のように書いておきます。

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

この方が美しいのです。わからない方は読みとばして下さい。行列

がわからなければ回転ができないというものではありません。
　行列が発明されるずっと以前から物体は回転していたのですから。

　さて前の図を逆に見ますと、次のようなことがわかります。
　いま図①の P(x,y) を角度$-\theta$だけ回転させて P′(X, Y) に移すための式が次式でした。



図①

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

　この図では角度を$-\theta$にとっているので、点 P(x,y)と P′(X, Y) を入れかえて、図②のように角度θだけ回転させることを考えましょう。
　そうするとθを$-\theta$におきかえて、

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

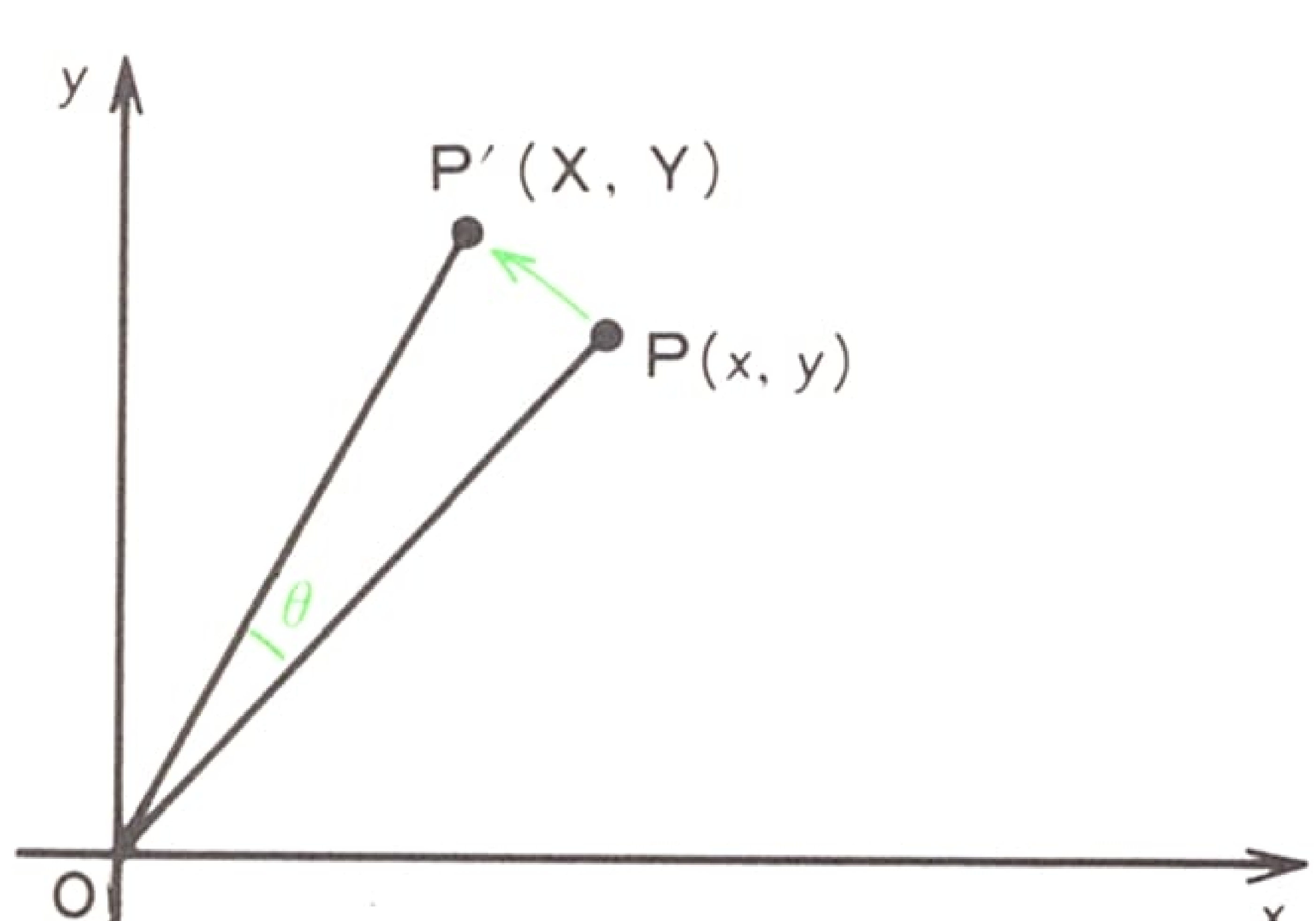
●ここで
$\cos(-\theta) = \cos\theta$
$\sin(-\theta) = -\sin\theta$
を使っています。

　となります。成分に分けて書くと次のようになります。

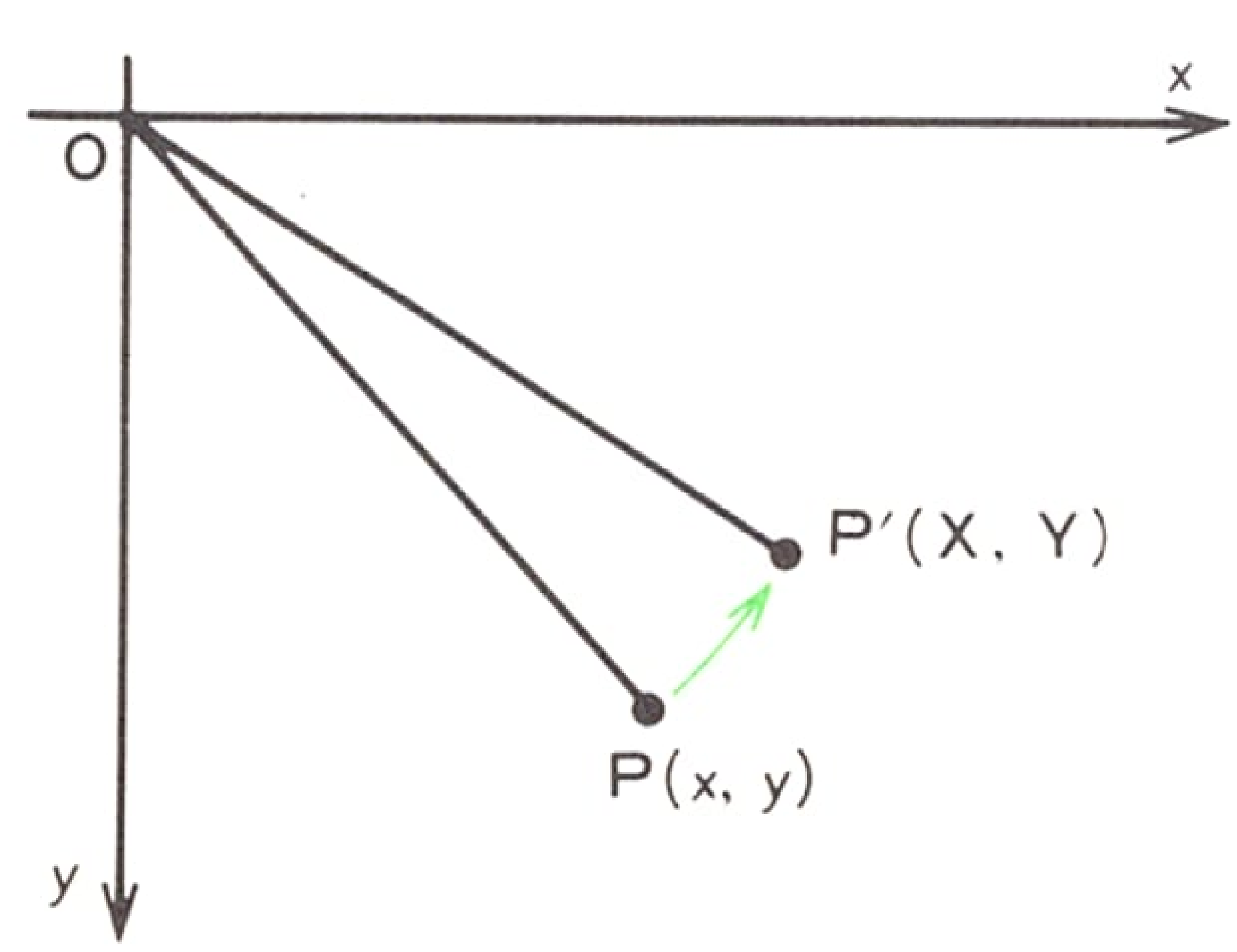
$$X = x\cos\theta - y\sin\theta$$
$$Y = x\sin\theta + y\cos\theta$$

●この辺はあまり真剣に考えると頭がおかしくなります。

ＰＣ－８８０１の画面では、y 軸がさかさまで、再び回転角を逆にして考える必要がありますが、見かけの回転角は上で考えたのと同じことになります。(図③)
　座標軸を考えに入れなければ図形を 20° まわそうと考えていれば、画面でも 20° まわることになります。

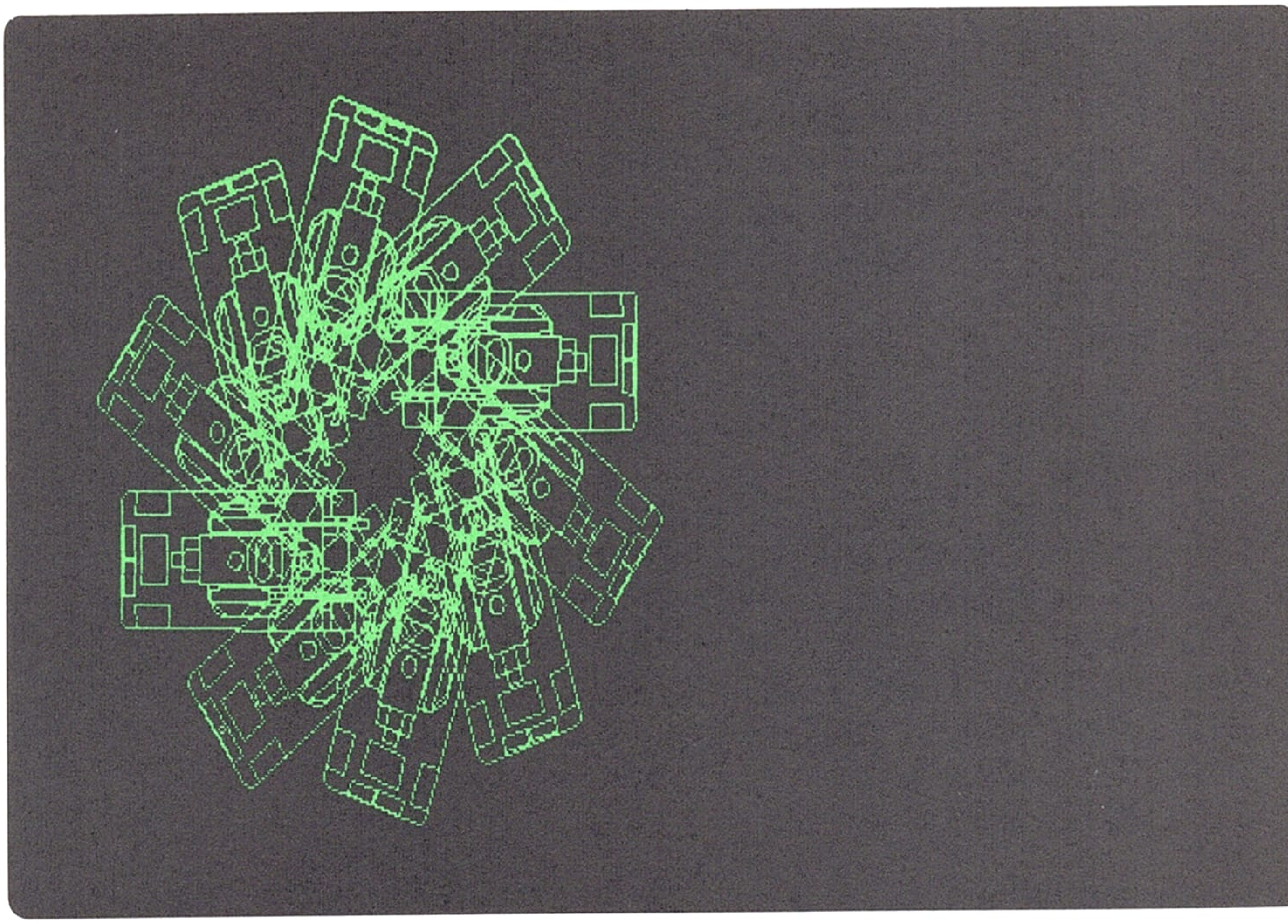


図②



図③

理論はこのくらいにして実技に入っておきましょう。最初に車体と上部構造物を同時に回転させてやることにしましょう。下図では 18° おきではなく、36° おきに表示させています。回転の中心は原点です。そのためWINDOWを広げています。



次ページにプログラムを示します。このプログラムを走らせる時には車体部分のデータと上部構造物のデータを完全にして下さい。行番号1000および行番号2000からのDATA文は3・4節のDATA 1およびDATA 2を使ってください。プログラム4－2では、ほんの一部しか表示されません。

☞ 8 (P.221)

●プログラム 4−2●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WINDOW (-1280,-800)-(1280,800)
115 VIEW (0,0)-(639,199)                      ← ウィンドウを広くとる
120 CONSOLE 0,25,0,1:WIDTH 80,25
130 COLOR 7,0,0,7
140 COL=7
150 FOR K=1 TO 20
160 RAD=-3.14159*K*18/180                     ← 回転角を決める
170 RESTORE 1000 :GOSUB *GDRAW                ← 車体部分を描く
180 RESTORE 2000 :GOSUB *GDRAW                ← 上部構造物を描く
190 NEXT K
200 END
210 '
220 *GDRAW
230 READ X,Y,F
240 IF F=0 THEN RETURN
250 W=X*COS(RAD)+Y*SIN(RAD)
260 Y=-X*SIN(RAD)+Y*COS(RAD)
270 X=W
280 IF F>1 THEN PSET(X,Y),COL:GOTO 300
290 LINE(XOLD,YOLD)-(X,Y),COL
300 XOLD=X:YOLD=Y
310 GOTO 230
320 '
1000 DATA 43,202,2
1010 DATA 43,168,1
1020 DATA 44,165,1                            ← 車体部分のデータ
1030 DATA 46,164,1
1040 DATA 464,165,1
1050 DATA 0,0,0
1060 '
2000 DATA 144,233,2
2010 DATA 155,224,1
2020 DATA 161,222,1
2030 DATA 224,221,1                           ← 上部構造物のデータ
2040 DATA 229,223,1
2050 DATA 235,226,1
2060 DATA 334,227,1
2070 DATA 0,0,0
```

●データ文は未完成です。3・4節のＤＡＴＡ１、ＤＡＴＡ２を使ってください。

4・2 車体上の上部構造物だけを回転させる

☞ 9 (P.222)

車体全体が回転するのも面白いですが、上部構造物だけを回転させてみるのも面白いことです。それにはどうしたらよいでしょう。上部構造物だけを回転させるには車体と上部構造物を分離してしまいます。

まず上部構造物の中心の座標をさがします。本物ではどこに中心がおかれるのか良くわからないのですが、大体の判断で次のように決めました。

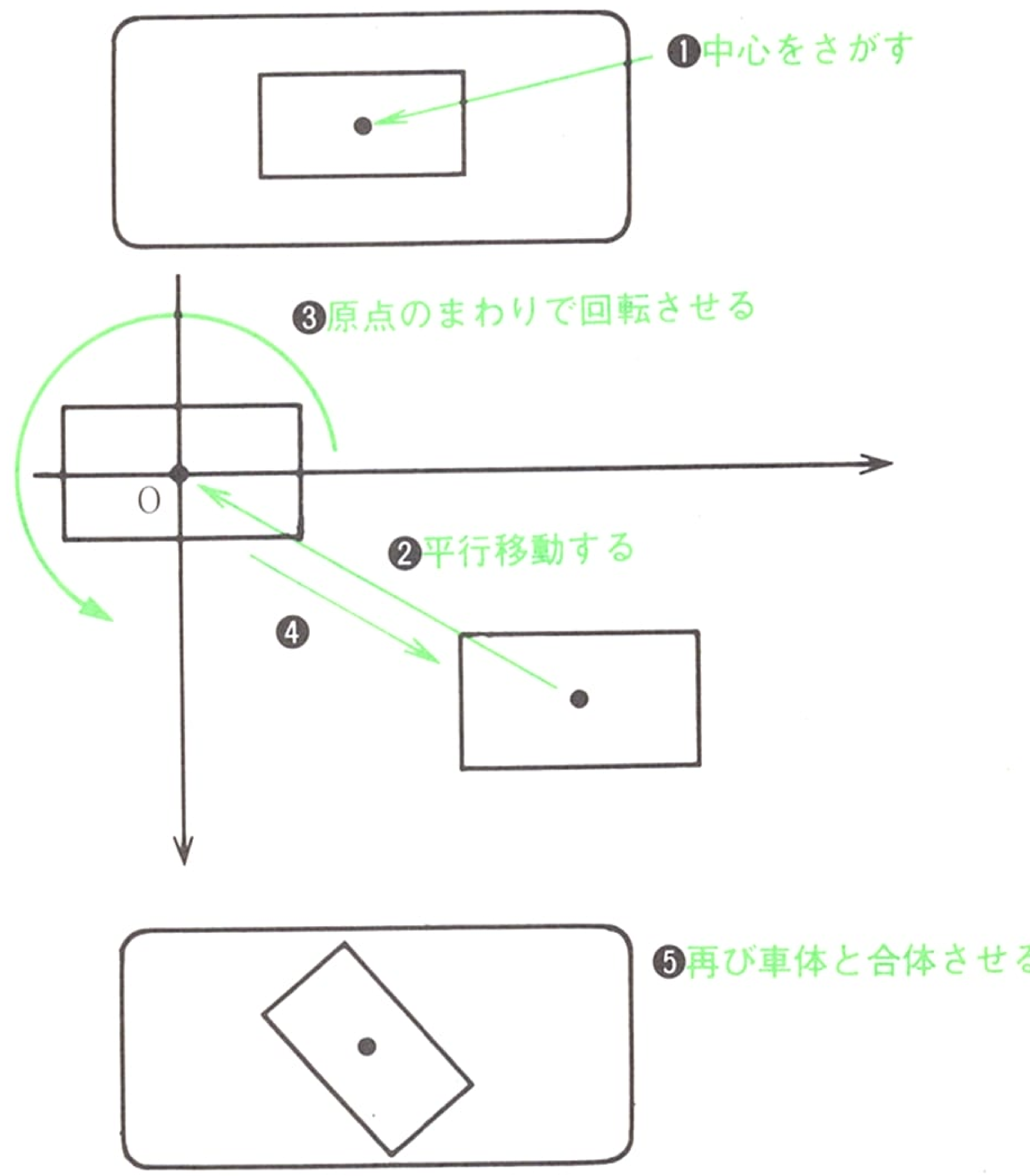
(223,259)

次にこの中心が座標軸の原点に来るように平行移動します。

そして原点のまわりで上部構造物を指定された角度だけ回転させます。

これを再び車体と合体させるという手続きでおしまいです。

プログラムはそれほど難しくはないと思います。



表示結果は全ては示してありません。真黒に近くなり見にくいので途中で止めてあります。

●プログラム 4—3●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WINDOW (0,0)-(639,399):VIEW (0,0)-(639,199)
120 CONSOLE 0,25,0,1:WIDTH 80,25
130 COLOR 7,0,0,7
140 COL=7
150 GOSUB *GDRAW1        ← 車体部分を描く
160 FOR K=0 TO 20  ┐
170 GOSUB *GDRAW2  ├── 上部構造物を回転させる
180 NEXT K         ┘
```

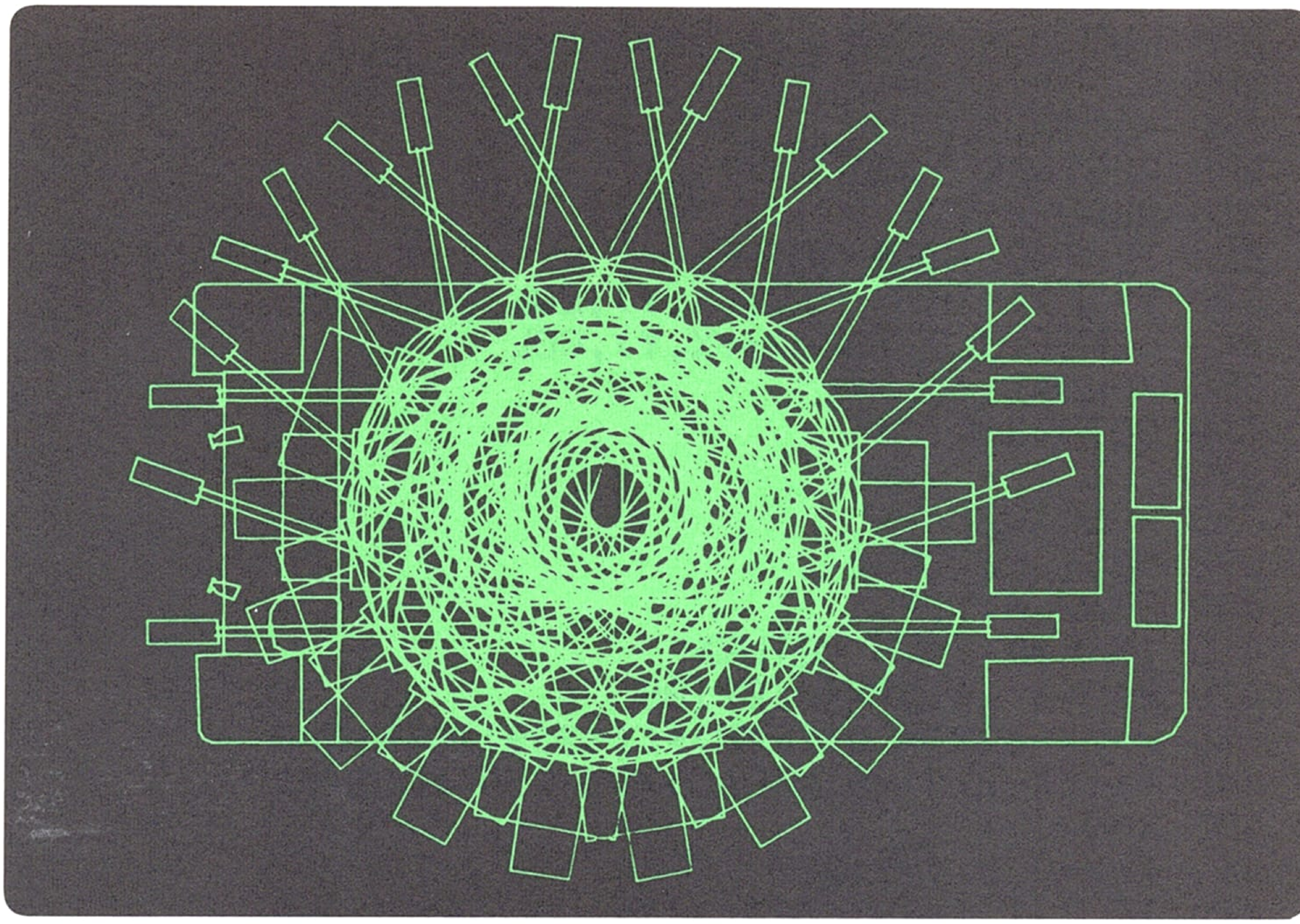
```
190 END
200 '
210 *GDRAW1
220 RESTORE 1000
230 READ X,Y,F
240 IF F=0 THEN RETURN
250 IF F>1 THEN PSET(X,Y),COL:GOTO 270
260 LINE(XOLD,YOLD)-(X,Y),COL
270 XOLD=X:YOLD=Y
280 GOTO 230
290 '
300 *GDRAW2
310 RESTORE 2000
320 READ X,Y,F
330 IF F=0 THEN RETURN
340 RAD=3.14159*K*18/180
350 W=(X-223)*COS(RAD)+(Y-259)*SIN(RAD)
360 Y=-(X-223)*SIN(RAD)+(Y-259)*COS(RAD)
370 X=W
380 X=X+223:Y=Y+259
390 IF F>1 THEN PSET(X,Y),COL:GOTO 410
400 LINE(XOLD,YOLD)-(X,Y),COL
410 XOLD=X:YOLD=Y
420 GOTO 320
430 '
1000 DATA 43,202,2
1010 DATA 43,168,1
1020 DATA 44,165,1
1030 DATA 46,164,1
1040 DATA 464,165,1
1050 DATA 0,0,0
1060 '
2000 DATA 144,233,2
2010 DATA 155,224,1
2020 DATA 161,222,1
2030 DATA 224,221,1
2040 DATA 229,223,1
2050 DATA 235,226,1
2060 DATA 334,227,1
2070 DATA 0,0,0
```

（210〜280）車体部分を描くサブルーチン

（340）← 回転角を決める

●データ文は未完成です。3・4節のＤＡＴＡ１、ＤＡＴＡ２を使ってください。

プログラム４－３の実行例を示します。



さらに上部構造物を左右にふりまわすテクニックを使ってみたのが次のプログラムです。SCREEN 文で画面を切り変えています。

●プログラム 4−4●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WINDOW (0,0)-(639,399):VIEW (0,0)-(639,199)
120 WIDTH 80,25:CONSOLE 0,25,0,1
130 COLOR 7,0,0,7
140 COL=7
150 SCREEN 0,0
160 GOSUB *GDRAW1
170 FOR K=0 TO 2
180 SCREEN 1,,K,2^K
190 GOSUB *GDRAW2
200 NEXT K
210 ′
220 FOR K=1 TO 20
230 SCREEN 1,,,1:FOR I=1 TO 300:NEXT I
240 SCREEN 1,,,2:FOR I=1 TO 300:NEXT I
250 SCREEN 1,,,4:FOR I=1 TO 300:NEXT I
```

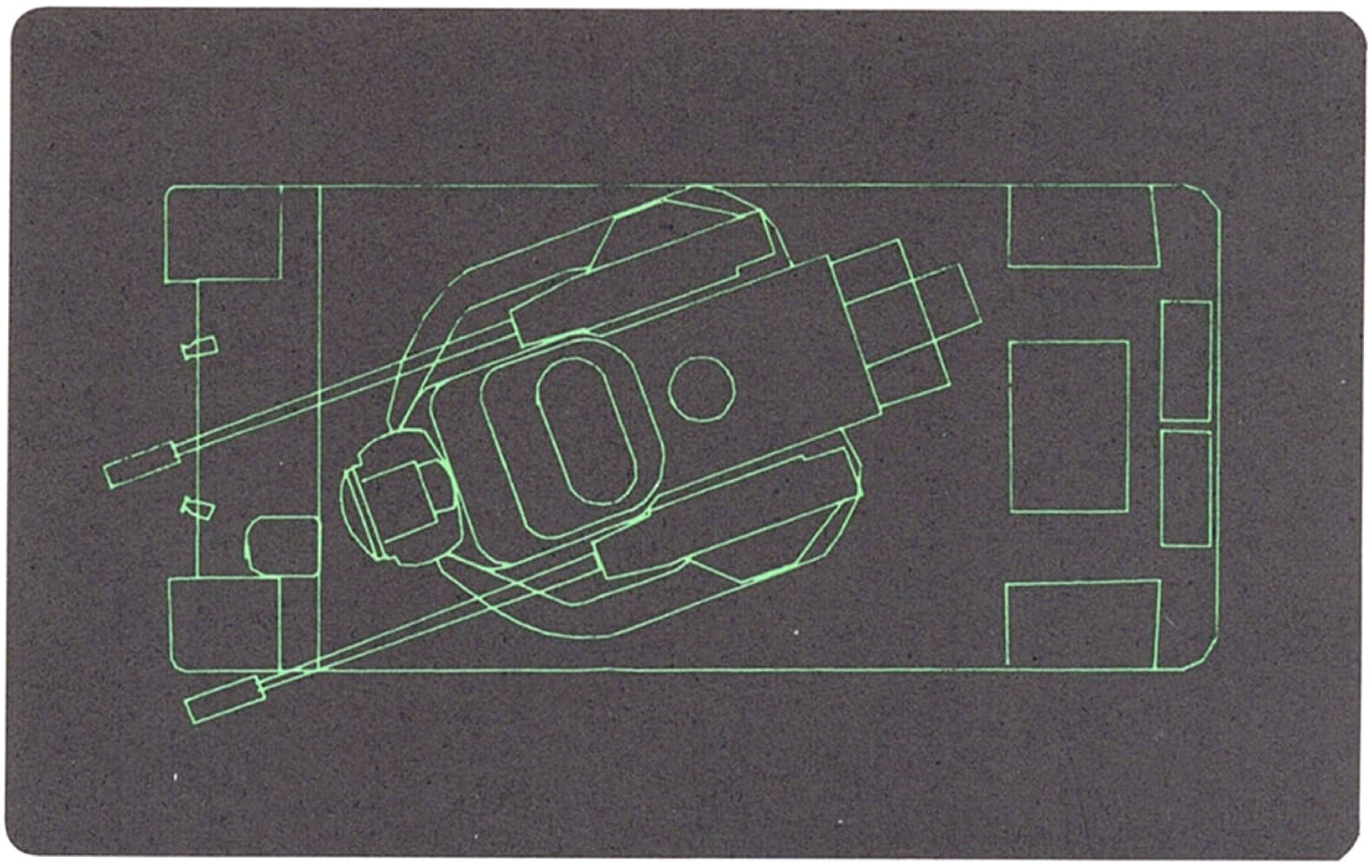
（160行）車体部分を描く

（170〜200行）プレーン1、2、3に上部構造物を描く

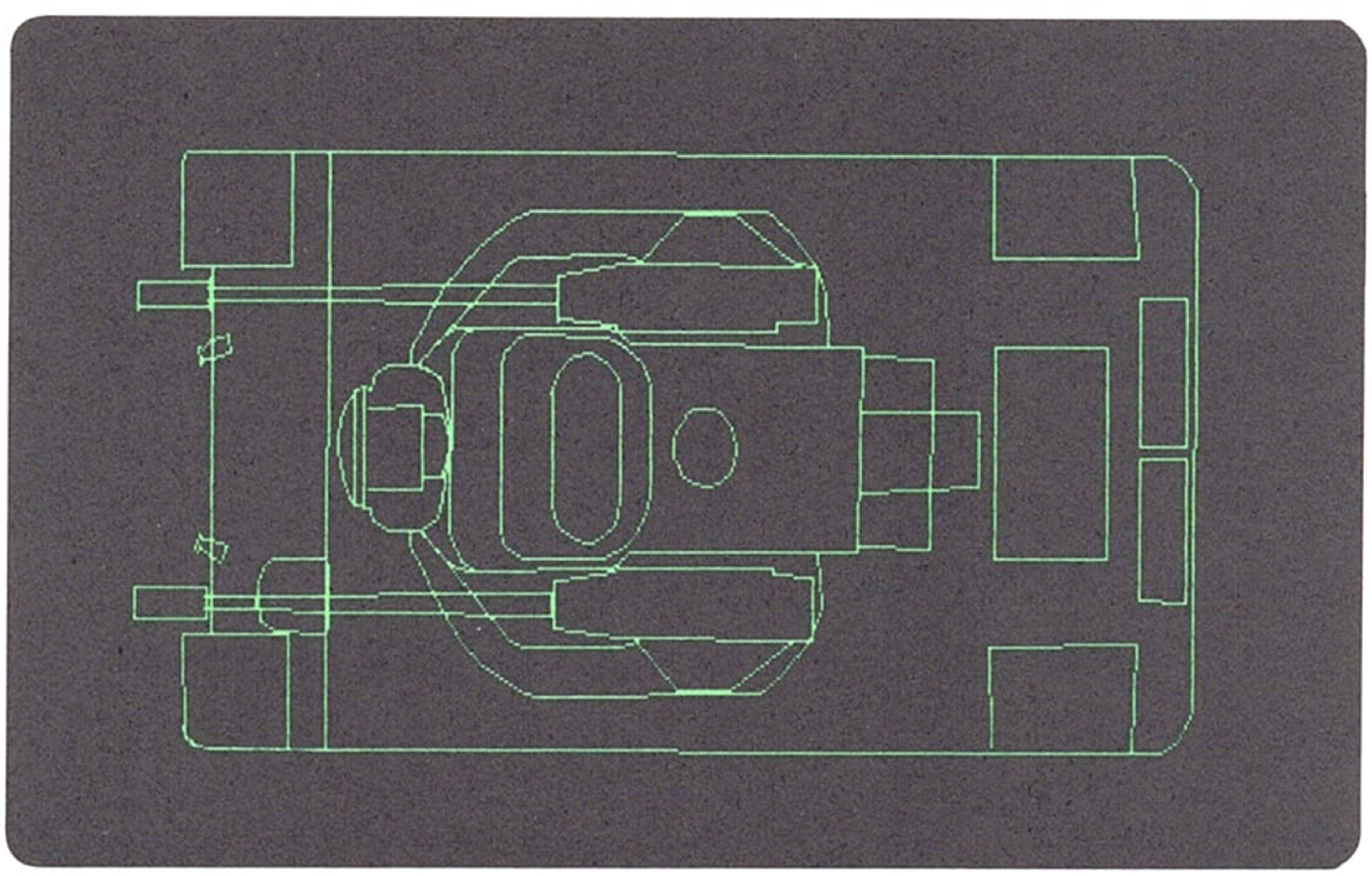
（230〜250行）ディスプレイページを切り換えて上部構造物をふりまわす

```
260 NEXT K
270 END
280 '
290 *GDRAW1
300 RESTORE 1000
310 READ X,Y,F
320 IF F=0 THEN RETURN
330 IF F>1 THEN PSET(X,Y),COL:GOTO 350
340 LINE(XOLD,YOLD)-(X,Y),COL
350 XOLD=X:YOLD=Y
360 GOTO 310
370 '
380 *GDRAW2
390 RESTORE 2000
400 READ X,Y,F
410 IF F=0 THEN RETURN
420 RAD=3.14159*K*18/180
430 W=(X-223)*COS(RAD)+(Y-259)*SIN(RAD)
440 Y=-(X-223)*SIN(RAD)+(Y-259)*COS(RAD)
450 X=W
460 X=X+223:Y=Y+259
470 IF F>1 THEN PSET(X,Y),COL:GOTO 490
480 LINE(XOLD,YOLD)-(X,Y),COL
490 XOLD=X:YOLD=Y
500 GOTO 400
510 '
1000 DATA 43,202,2
1010 DATA 43,168,1
1020 DATA 44,165,1
1030 DATA 46,164,1
1040 DATA 464,165,1
1050 DATA 0,0,0
1060 '
2000 DATA 144,233,2
2010 DATA 155,224,1
2020 DATA 161,222,1
2030 DATA 224,221,1
2040 DATA 229,223,1
2050 DATA 235,226,1
2060 DATA 334,227,1
2070 DATA 0,0,0
```

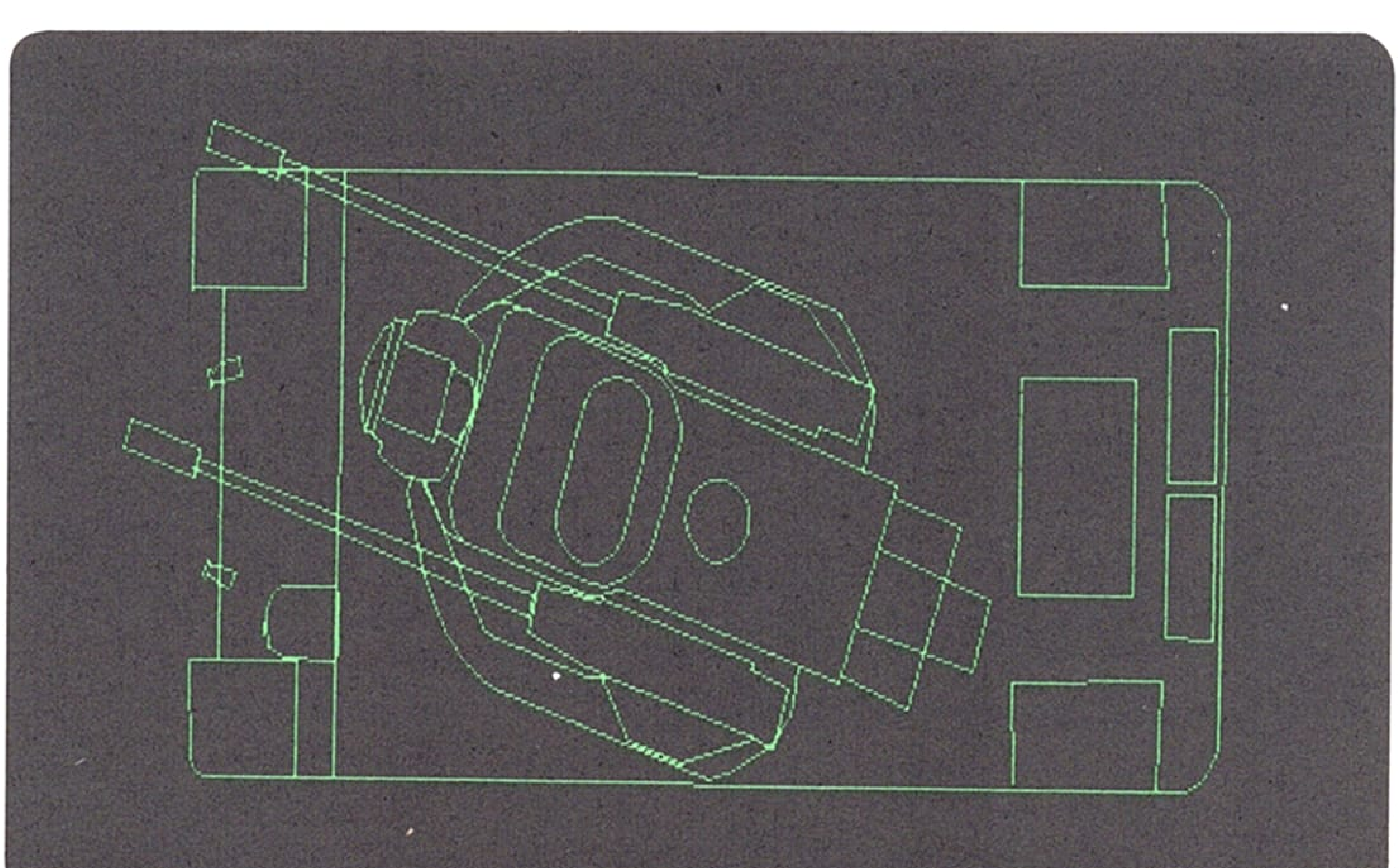
●データ文は未完成です。3・4節のＤＡＴＡ１、ＤＡＴＡ２を使ってください。



SCREEN 1,,,1



SCREEN 1,,,2



SCREEN 1,,,4

4・3 物体を拡大・縮小する

ある図形を拡大したり縮小することは縦横の比率を変えるだけですから、原理的にはそんなに難しいことはありません。

$$X = Sx \times x$$
$$Y = Sy \times y$$

で良いのです。おもしろいのは行列表現です。まともに書くと、

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} Sx & 0 \\ 0 & Sy \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

ですが、次のように工夫すると、もっと美しくなります。

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} Sx & 0 & 0 \\ 0 & Sy & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

2次元図形の平行移動・回転・拡大縮小はすべて3×3の行列で統一的に表現できることがわかります。これは大変不思議でおもしろい性質です。ここでは変換行列の性質を特に論ずるつもりはありませんが、興味のある読者は進んだ本を勉強して下さい。

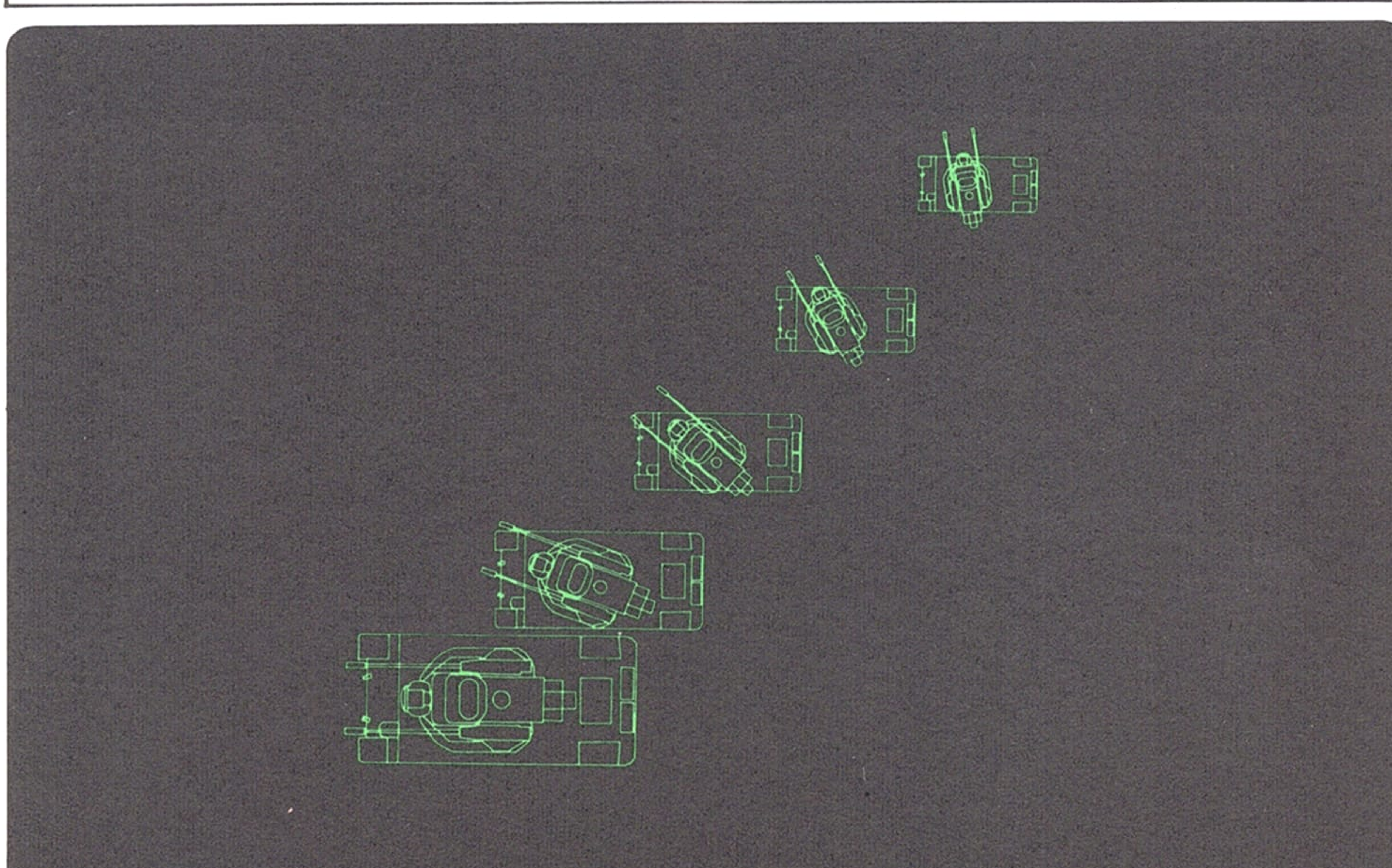
☞ 10 (P.223)

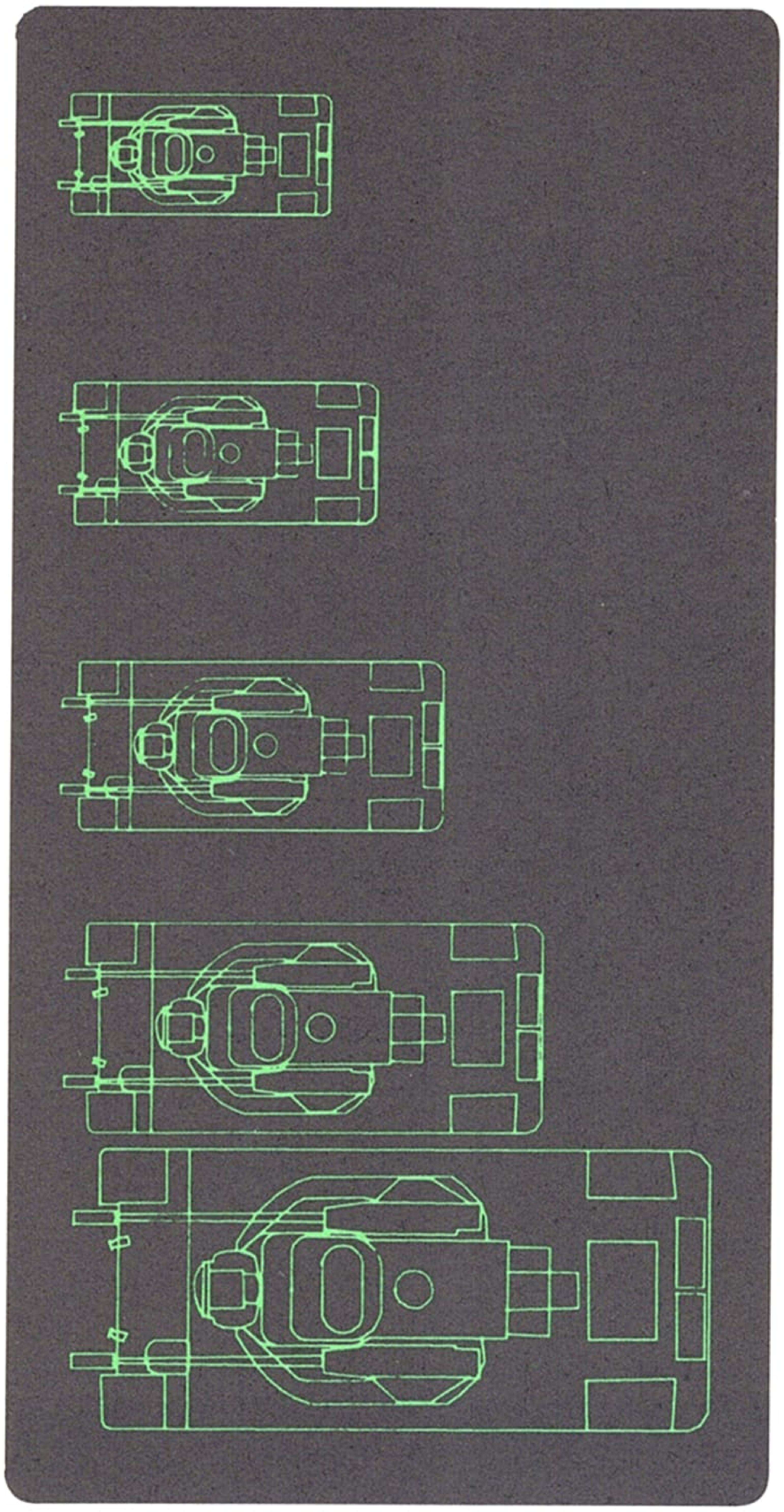
プログラム例を作ってみました。拡大・縮小は常に原点を基準に行なわれることに注意して下さい。出力例にはプリンタを使わず、プロッタを使いましたので、本文中のプログラムの出力とは少し違う所があります。いろいろな出力を3点ほど用意してあります。参考になさって下さい。

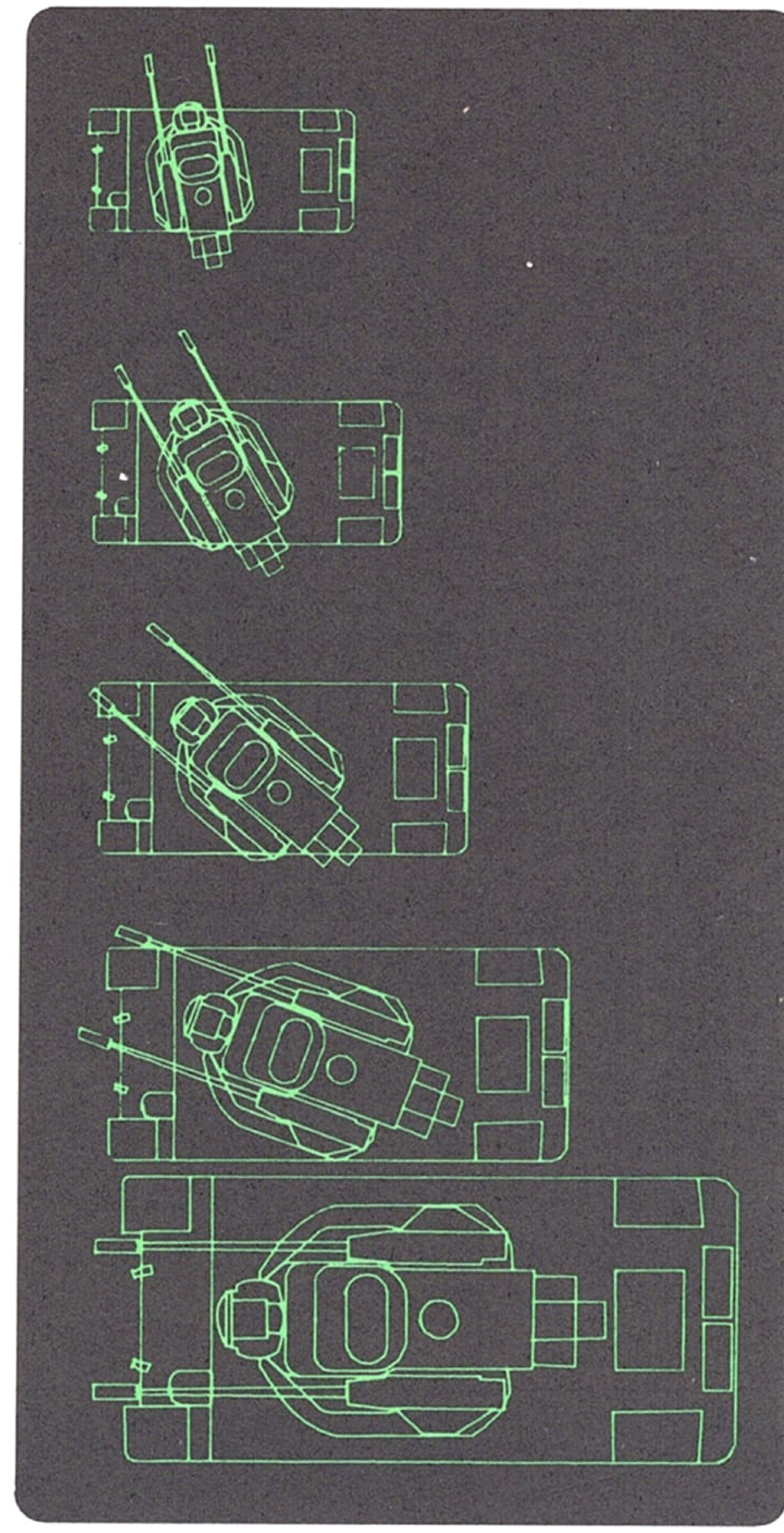
●プログラム4－5●

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WINDOW (0,0)-(639,399):VIEW (0,0)-(639,199)
120 CONSOLE 0,25,0,1:WIDTH 80,25
130 COLOR 7,0,0,7
140 COL=7
150 '
160 FOR K=3 TO 7
170 RESTORE 1000 :GOSUB *GDRAW1
180 RESTORE 2000 :GOSUB *GDRAW1
190 NEXT K
200 END
210 '
```

```
220 *GDRAW1
230 READ X,Y,F
240 X=X/K:Y=Y/K+230-60*(K-3)
250 IF F=0 THEN RETURN
260 IF F>1 THEN PSET(X,Y),COL:GOTO 280
270 LINE(XOLD,YOLD)-(X,Y),COL
280 XOLD=X:YOLD=Y
290 GOTO 230
300 '
1000 DATA 43,202,2
1010 DATA 43,168,1
1020 DATA 44,165,1
1030 DATA 46,164,1
1040 DATA 464,165,1
1050 DATA 0,0,0
1060 '
2000 DATA 144,233,2
2010 DATA 155,224,1
2020 DATA 161,222,1
2030 DATA 224,221,1
2040 DATA 229,223,1
2050 DATA 235,226,1
2060 DATA 334,227,1
2070 DATA 0,0,0
```







4・4 まとめ

これで２次元グラフィックスの章は終りです。いとも簡単に複雑なことができてしまうのに驚かれたでしょう。ＰＣ-８８０１のVRAMが現在の数倍あればもっといろいろなことができるのに残念だと思います。

私が初めて使ったマイコンのＴＫ80はＲＡＭの容量がわずかに512バイトでした。もちろんVRAMなどはついていませんでした。それから７年間でＰＣ-８８０１はVRAMが48キロバイト、他にRAMが64キロバイトになりました。合計すると約200倍の怪物にふくれ上がりました。

しかし、慣れはおそろしいものです。これでも不満で、VRAMだけで１メガバイトあっても使い切れるし、不足かもしれないと思っているのです。何とものすごい時代ではありませんか。いまのＰＣ-８８０１にはグラフィックス用の演算機構や演算素子はついていませんが、そうしたものが強く求められており、実装されるのもそう遠い日のことではないと思います。リアルタイムのシュミレーションをするにはどうしても必要だからです。

さて次章ではいよいよ３次元グラフィックスを勉強することにしましょう。

第5章

3次元グラフィックスの世界



5·0　3次元グラフィックスの特徴

3次元グラフィックスのことを3Dグラフィックスともいいます。DはDimension（次元）の略です。

3次元グラフィックスにおいては、2次元グラフィックスで勉強したテクニックをそのまま使うことができます。ですから平行移動や回転や拡大縮小は2次元の場合に勉強したことがらを、ほぼそのまま拡張することができます。それは次の節でまとめて示しておきますが、3次元グラフィックスが2次元グラフィックスと基本的にどう違うかをこの節では良く理解しておくことにしましょう。

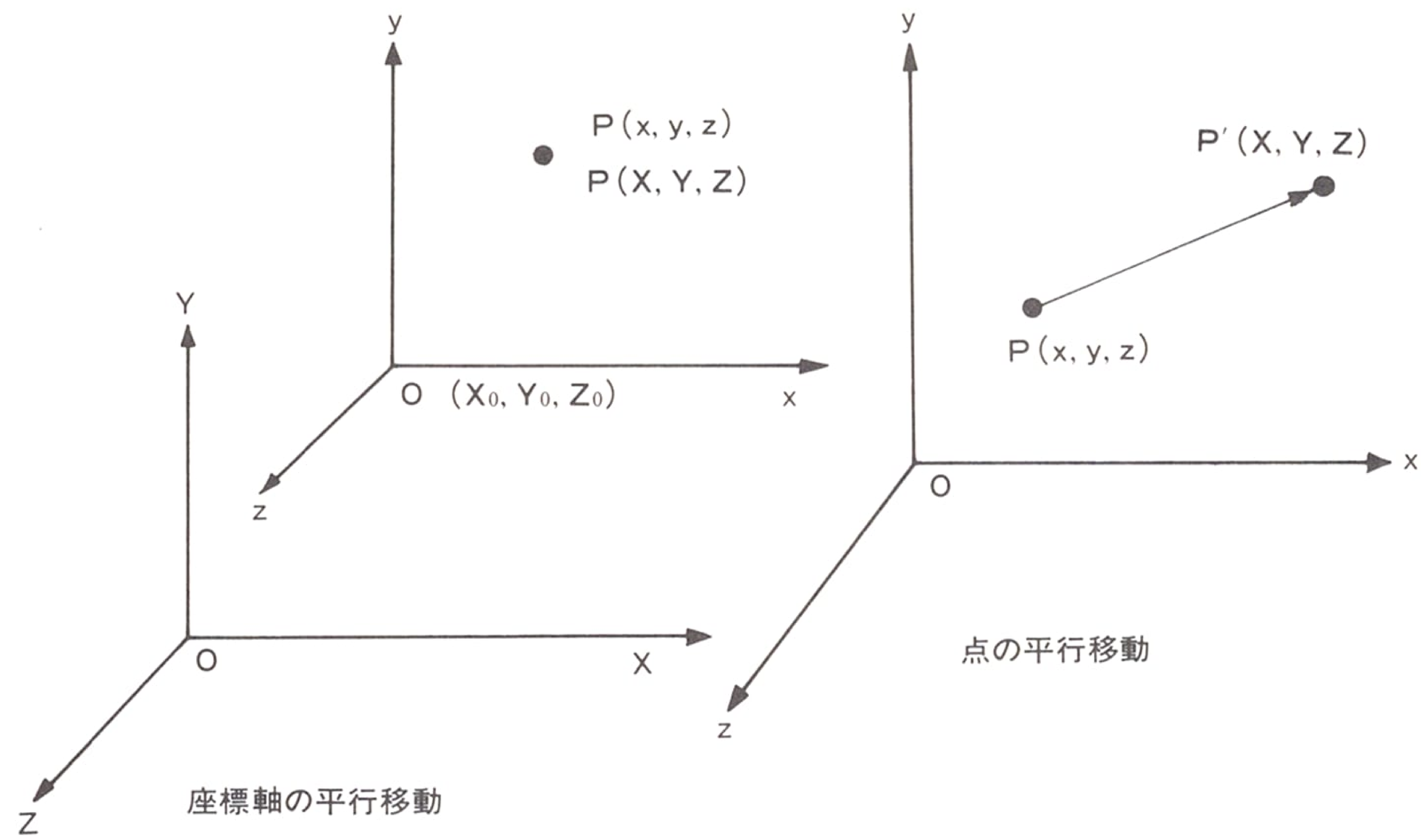
2次元グラフィックスにおいては、2次元の図形を2次元のCRT画面に表示すれば話はすむのでした。ところが3次元グラフィックスでは3次元の図形を2次元のCRT画面に表示しなければなりません。ここが3次元グラフィックスの特徴であり、難しいところです。

3次元の図形を2次元でどう表現するか、これは大変難しい問題であると言えます。3次元の図形では縦・横・高さの3つの軸、すなわちX, Y, Z軸があるのに、CRT画面では縦と横、つまり、X, Y軸の2つしかありません。ですから、3次元物体を平行移動したり、回転したり、拡大縮小したりする操作の他に、3次元図形を2次元で表示するための特別な操作が必要となります。行列表示による見かけの難しさにもかかわらず、本質的に難しく、わかりにくいのは後者の3次元図形を2次元で表示するためのテクニックです。

アップル用のコンピュータグラフィックスの本を書いたR.E.Myers氏の本の中にあったことで、上手なたとえだと感心した記述があります。それは本章で勉強する3次元グラフィックスはスーパーマンのX線透視であるということです。スーパーマンは鉛の物体以外は全てX線透視できて、ふつうの人々にはうらやましいと思えるでしょうが、何でも透視できることはかえって困ることがあるのです。X線透視の世界は非常に混乱させる部分があり、ふつうの人々が眼で見る通りに表示するためのテクニックがあります。これを陰線消去のテクニックといいます。これはオリエンテーションという向きづけをしたり、内積を計算したり、多重度を考慮しなくてはならないのでとり入れてありません。さらに最近のCADでは光の当たり方を考慮した陰影のつけ方まで処理できるようになっています。大変高級なテクニックで、これによって物体が実在感を持つようになります。これなども取り入れたかった材料でした。本章ではあまり欲張らずに、非常に基本的なテクニックをつみ重ねて、最後には宇宙船エンタープライズ号をCRT画面上に浮かび上がらせるようにしましょう。

5·1 3次元空間における平行移動

3次元空間における平行移動は簡単です。2次元の場合をそのまま拡張すればよいのです。今2つの座標系xyzとXYZがあって互いにX_0, Y_0, Z_0だけ平行移動した関係にあるとします。図を見て下さい。



座標軸の平行移動

点の平行移動

この時、2つの座標系の間には次のような関係があります。

$$X = x + X_0$$
$$Y = y + Y_0 \cdots\cdots\cdots\cdots ❶$$
$$Z = z + Z_0$$

これを行列表現すると次のようになります。

$$\begin{bmatrix} Y \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & X_0 \\ 0 & 1 & 0 & Y_0 \\ 0 & 0 & 1 & Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \cdots\cdots\cdots\cdots ❷$$

この関係は座標軸の平行移動から見方を変えて、個々の点の平行移動という立場から見直してみますと（x, y, z）にある点を（X, Y, Z）に平行移動するには❶のようにすればよいとも解釈できます。

平行移動用のサブルーチンは次のように作ることができます。

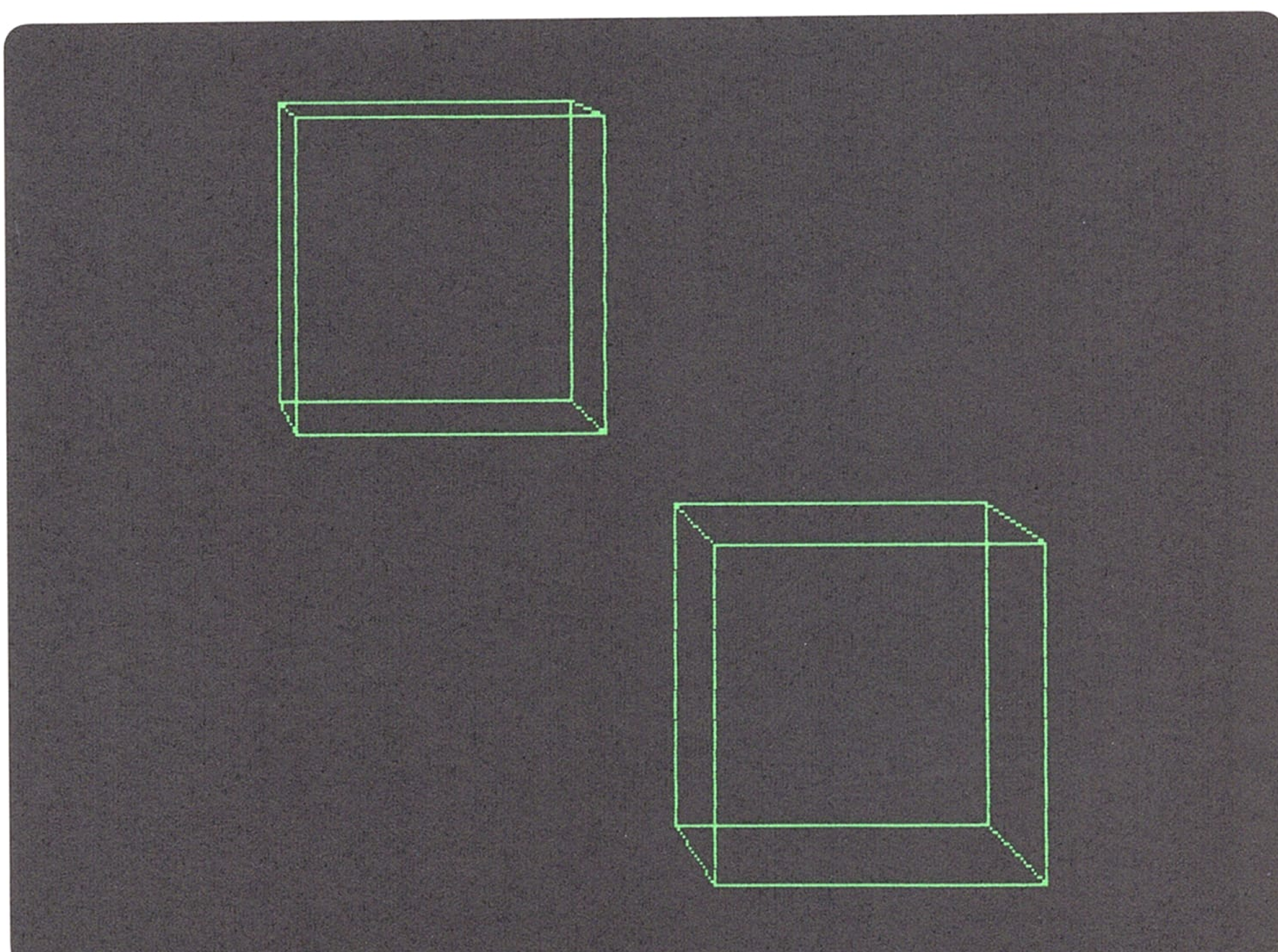
●プログラム5－1●

```
100 N=16
110 DIM P(N,3)  ←―― データ用配列
120 WIDTH 80,25:CONSOLE 0,25,0,1
130 SCREEN 0,3:CLS 3:SCREEN 0,0
140 WINDOW (0,0)-(320,200):VIEW (0,0)-(639,199)
150 PTH=1000:TZ=2000
160 '
170 GOSUB *DINP  ←―― データの読み込み
180 COL=2:GOSUB *DISP  ←―― 立体図形を画面に表示
190 X0=120:Y0=120:Z0=120:GOSUB *MOVE3D  ←―― 座標を平行移動する
200 COL=4:GOSUB *DISP  ←―― 平行移動した立体図形を
                            画面に表示する。
210 END
220 '
230 *DINP
240 FOR I=1 TO N
250 READ P(I,0),P(I,1),P(I,2),P(I,3)
260 NEXT I
270 RETURN
280 '
290 *DISP
300 FOR I=1 TO N
310 X=P(I,0):Y=P(I,1):Z=P(I,2):F=P(I,3)
320 IF F>1 THEN PSET(PTH*X/(TZ-Z),PTH*Y
/(TZ-Z)),COL:GOTO 350
330 ZOL=TZ-ZOLD:ZZ=TZ-Z
340 LINE(PTH*XOLD/ZOL,PTH*YOLD/ZOL)-(PT
H*X/ZZ,PTH*Y/ZZ),COL
350 XOLD=X:YOLD=Y:ZOLD=Z
360 NEXT I
370 RETURN
380 '
390 *MOVE3D
400 FOR I=1 TO N
410 P(I,0)=P(I,0)+X0
420 P(I,1)=P(I,1)+Y0
430 P(I,2)=P(I,2)+Z0
440 NEXT I
450 RETURN
460 '
470 DATA 100,100,100,2
480 DATA 100,200,100,1
490 DATA 200,200,100,1
```

```
500 DATA 200,100,100,1
510 DATA 100,100,100,1
520 DATA 100,100,200,1
530 DATA 200,100,200,1
540 DATA 200,200,200,1
550 DATA 100,200,200,1
560 DATA 100,100,200,1
570 DATA 100,200,100,2
580 DATA 100,200,200,1
590 DATA 200,200,100,2
600 DATA 200,200,200,1
610 DATA 200,100,100,2
620 DATA 200,100,200,1
```

このプログラムでは、立体図形をまずディスプレイに表示し、それをさらに前方へ平行移動させています。

実行結果を次に示します。



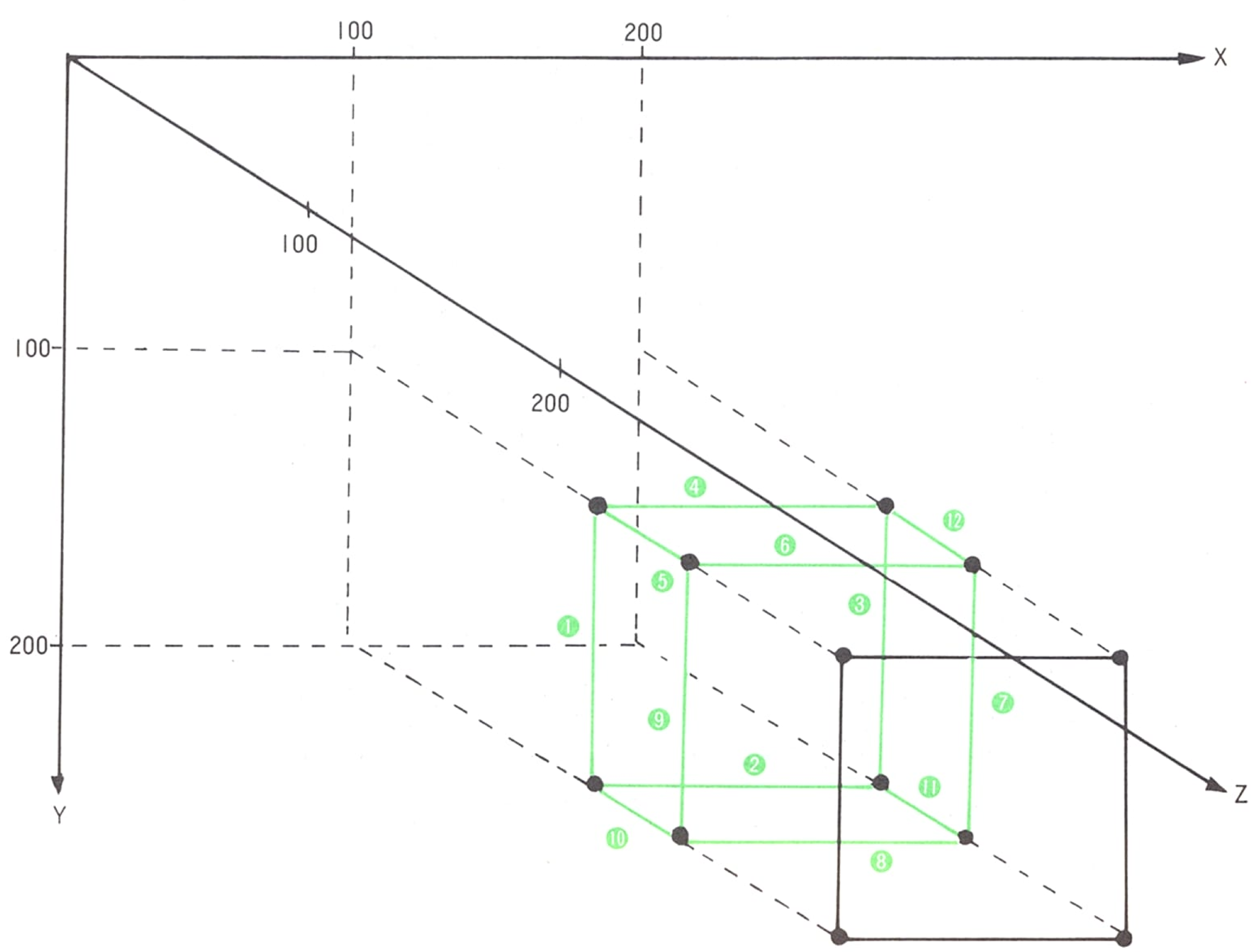
プログラム５－１のDATA文は立体図形の各頂点の座標とフラグをあらわしていますが、これをディスプレイ上に表示しますと手前の正方形の面は下の図の黒色の実線のようになります。

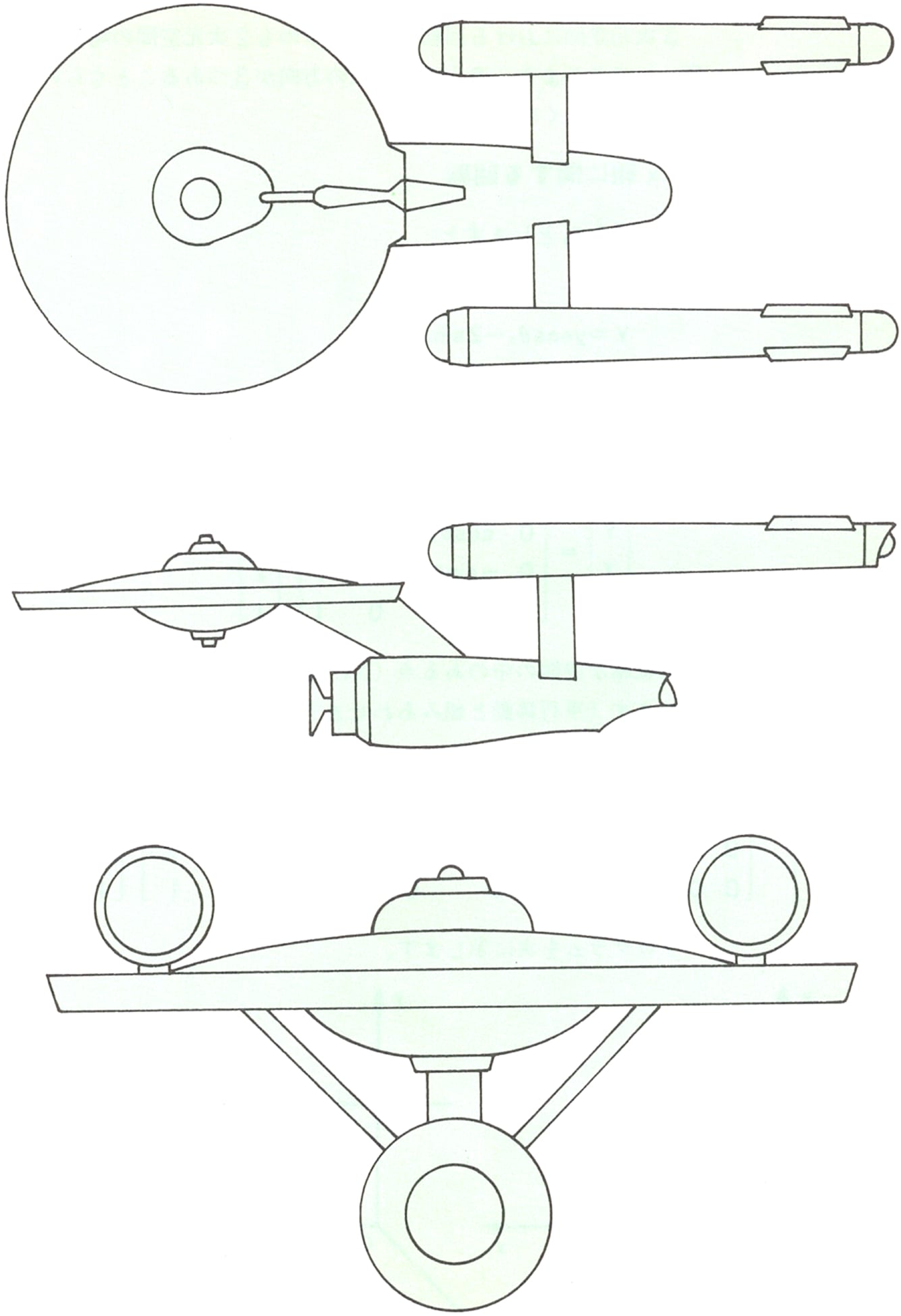
プログラムではＺ軸座標が前の点から変化するとき、つまり、行番号510、行番行520のデータ処理の際、行番号340のLINE文で後方に退くように調整されているため、実行結果は緑色の図形のようになり、より３次元的な感じが表現できるわけです。

このように３次元グラフィックスといっても、前述したように、Ｚ軸が実際にあるわけではなく、ＸＹ座標上でいかに拡大縮小あるいは回転などを行うかということになります。

このプログラムでは立体図形の各稜線は緑色の数字で示した順序に引かれます。これがまずディスプレイに表示され、さらに行番号190、200行で前方へ移動させているわけです。

このような操作は大変重要です。なぜなら、我々はエンタープライズ号を部分品にわけて建造し、これを合体させて作ろうとしているのですから、一般的には別々の所で作られた船体部分を３次元空間の中で平行移動して合体させることになるからです。





5・2　3次元空間における回転

3次元空間における回転を記述するのも2次元空間の場合とほぼ同様にしてできます。違うのは回転の方向が3つあることぐらいです。図❶をごらんください。

1　x軸に関する回転

回転角をθ_xとしますと、

$$X = x$$
$$Y = y\cos\theta_x - Z\sin\theta_x$$
$$Z = y\sin\theta_x + Z\cos\theta_x$$

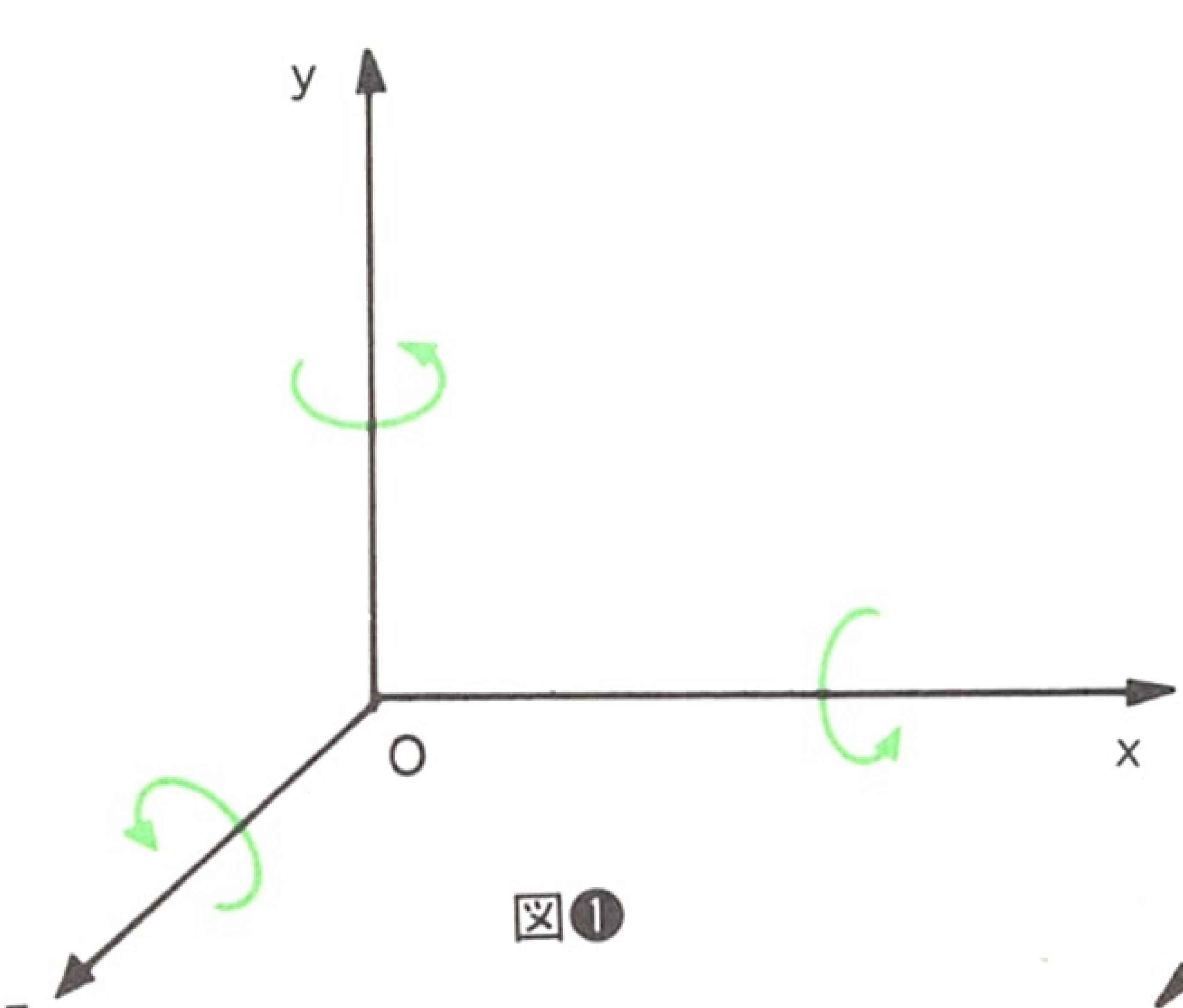
これを行列で表示しますと、次のようになります。

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta_x & \sin\theta_x & 0 \\ 0 & -\sin\theta_x & \cos\theta_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

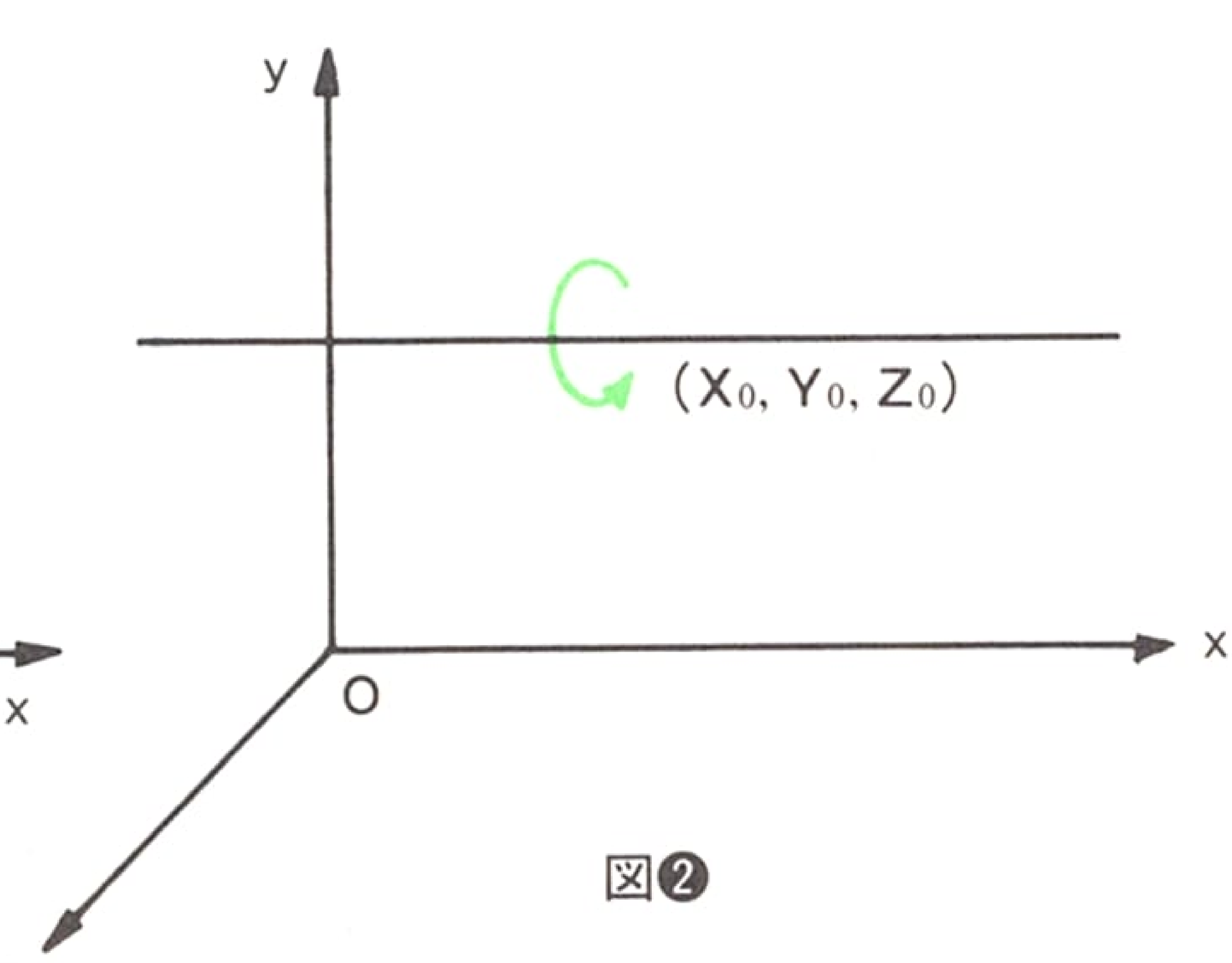
回転軸が空間の中のある点（X_0, Y_0, Z_0）を通る場合には次のようになります（平行移動と組みあわせます）。

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & X_0 \\ 0 & 1 & 0 & Y_0 \\ 0 & 0 & 1 & Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta_x & \sin\theta_x & 0 \\ 0 & -\sin\theta_x & \cos\theta_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -X_0 \\ 0 & 1 & 0 & -Y_0 \\ 0 & 0 & 1 & -Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

プログラムを次に示します。



図❶

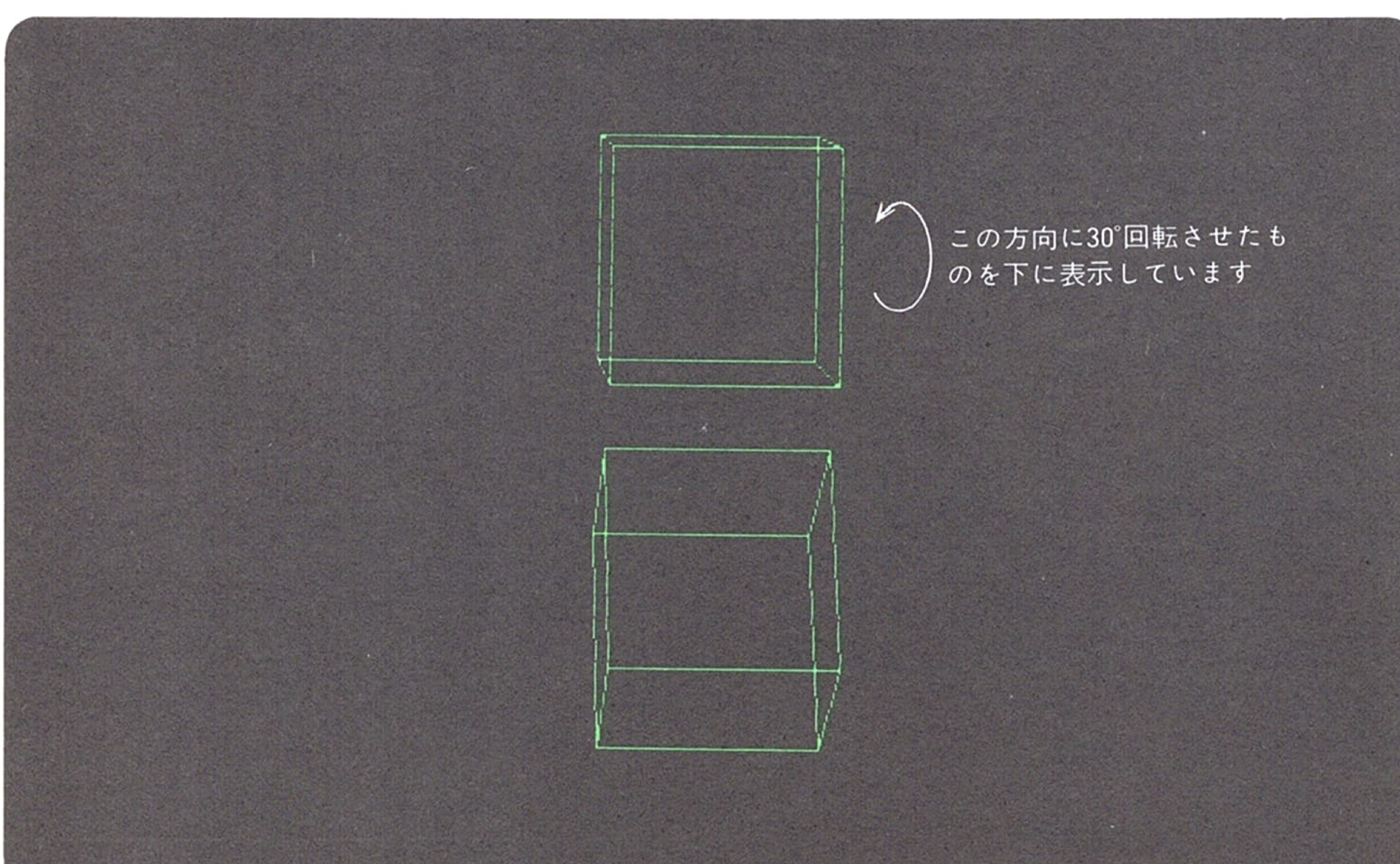


図❷

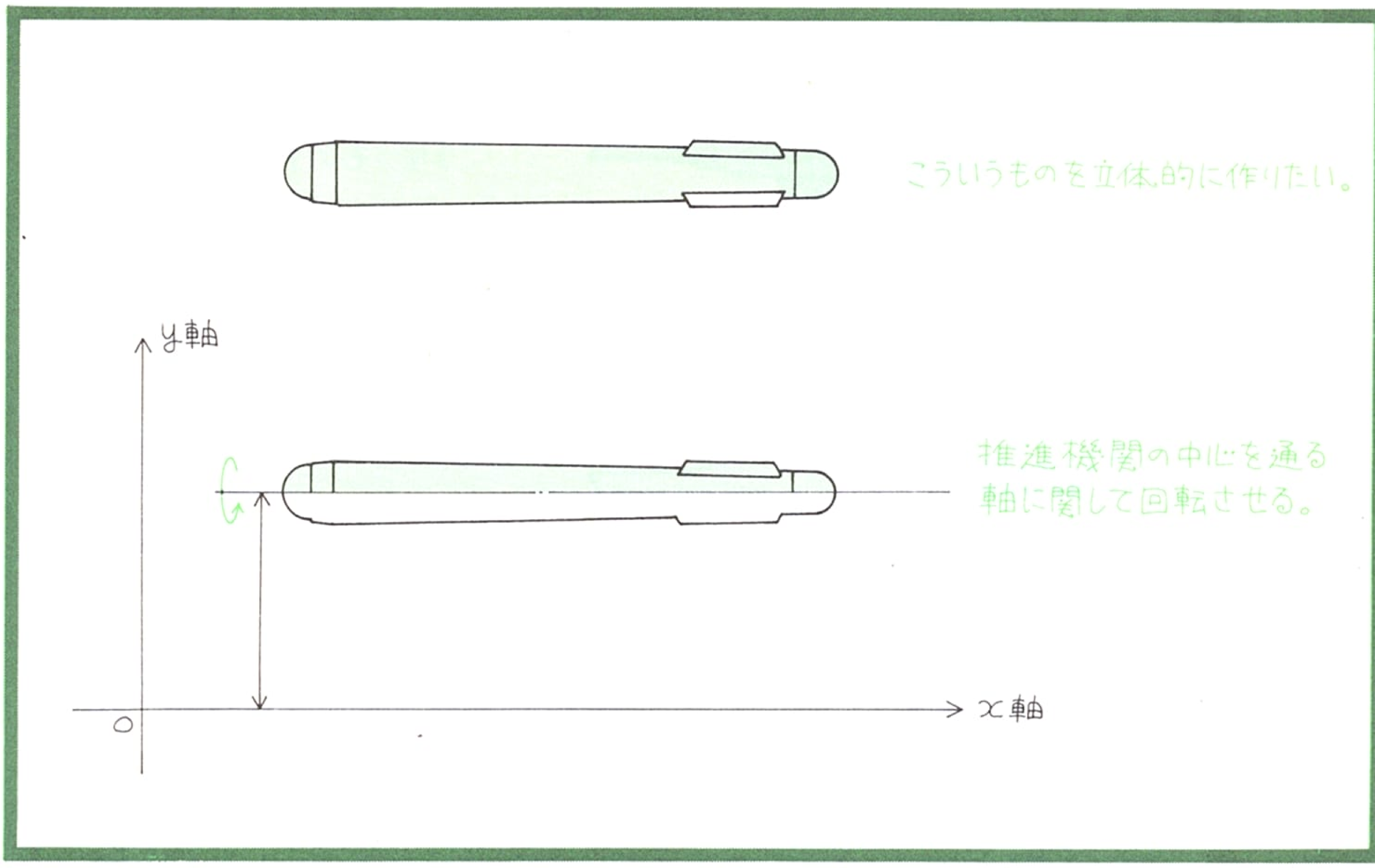
●プログラム5－2●

```
100 N=16
110 DIM P(N,3)
120 WIDTH 80,25:CONSOLE 0,25,0,1
130 SCREEN 0,3:CLS 3:SCREEN 0,0
140 WINDOW (0,0)-(320,200):VIEW (0,0)-(639,199)
150 PTH=1000:TZ=2000
160 '
170 GOSUB *DINP
180 COL=2:GOSUB *DISP
190 X0=0:Y0=-150:Z0=-150:GOSUB *MOVE3D
200 TX=2*3.14159*30/360:GOSUB *XROTATE
210 X0=0:Y0=300:Z0=150:GOSUB *MOVE3D
220 COL=4:GOSUB *DISP
230 END
240 '
250 *DINP
260 FOR I=1 TO N
270 READ P(I,0),P(I,1),P(I,2),P(I,3)
280 NEXT I
290 RETURN
300 '
310 *DISP
320 FOR I=1 TO N
330 X=P(I,0):Y=P(I,1):Z=P(I,2):F=P(I,3)
340 IF F>1 THEN PSET(PTH*X/(TZ-Z),PTH*Y
    /(TZ-Z)),COL:GOTO 370
350 ZOL=TZ-ZOLD:ZZ=TZ-Z
360 LINE(PTH*XOLD/ZOL,PTH*YOLD/ZOL)-(PT
    H*X/ZZ,PTH*Y/ZZ),COL
370 XOLD=X:YOLD=Y:ZOLD=Z
380 NEXT I
390 RETURN
400 '
410 *MOVE3D
420 FOR I=1 TO N
430 P(I,0)=P(I,0)+X0
440 P(I,1)=P(I,1)+Y0
450 P(I,2)=P(I,2)+Z0
460 NEXT I
470 RETURN
480 '
490 *XROTATE
```

```
500 FOR I=1 TO N
510      W=P(I,1)*COS(TX)-P(I,2)*SIN(TX)
520 P(I,2)=P(I,1)*SIN(TX)+P(I,2)*COS(TX)
530 P(I,1)=W
540 NEXT I
550 RETURN
560 '
570 DATA 100,100,100,2
580 DATA 100,200,100,1
590 DATA 200,200,100,1
600 DATA 200,100,100,1
610 DATA 100,100,100,1
620 DATA 100,100,200,1
630 DATA 200,100,200,1
640 DATA 200,200,200,1
650 DATA 100,200,200,1
660 DATA 100,100,200,1
670 DATA 100,200,100,2
680 DATA 100,200,200,1
690 DATA 200,200,100,2
700 DATA 200,200,200,1
710 DATA 200,100,100,2
720 DATA 200,100,200,1
```



船体の各部を作る時にはこの方法が応用されます。たとえば推進機関は次のようにして作られます。



2　y軸に関する回転

回転角をθ_yとしますと、

$$\begin{aligned} X &= x\cos\theta_y + Z\sin\theta_y \\ Y &= y \\ Z &= -x\sin\theta_y + Z\cos\theta_y \end{aligned}$$

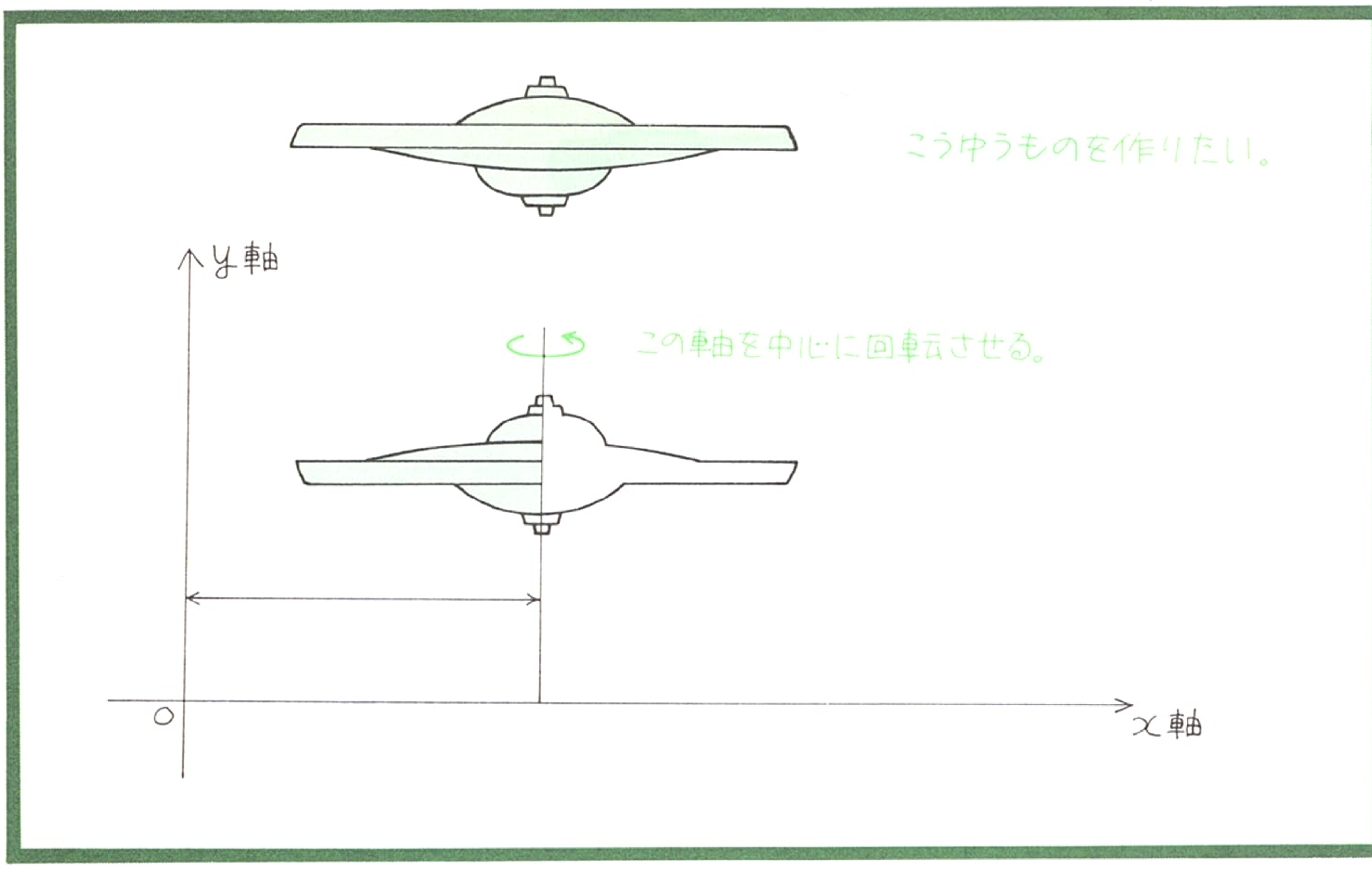
これを行列で表示しますと、次のようになります。

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta_y & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

回転軸が空間中のある点（X_0, Y_0, Z_0）を通る場合には次のようになります。

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & X_0 \\ 0 & 1 & 0 & Y_0 \\ 0 & 0 & 1 & Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta_y & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -X_0 \\ 0 & 1 & 0 & -Y_0 \\ 0 & 0 & 1 & -Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

このテクニックも大切です。艦橋部分はこのテクニックを応用して作れるからです。



3 z軸に関する回転

回転角をθ_zとしますと、

$$X = x\cos\theta_z - y\sin\theta_z$$
$$Y = x\sin\theta_z + y\cos\theta_z$$
$$Z = z$$

これを行列で表示しますと次のようになります。

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 & 0 \\ \sin\theta_z & \cos\theta_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

回転軸が空間中のある点（X_0, Y_0, Z_0）を通る場合は次のようになります。

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & X_0 \\ 0 & 1 & 0 & Y_0 \\ 0 & 0 & 1 & Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 & 0 \\ \sin\theta_z & \cos\theta_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -X_0 \\ 0 & 1 & 0 & -Y_0 \\ 0 & 0 & 1 & -Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

5・3 3次元空間における拡大と縮小

3次元空間における拡大と縮小は次式で実行されます。

$X = S_x \times x$

$Y = S_y \times y$

$Z = S_z \times z$

行列表示では次のようになります。

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

拡大縮小用のサブルーチンは次のように作ることができます。

●プログラム5－3●

```
100 N=16
110 DIM P(N,3)
120 WIDTH 80,25:CONSOLE 0,25,0,1
130 SCREEN 0,3:CLS 3:SCREEN 0,0
140 WINDOW (0,0)-(320,200):VIEW (0,0)-(639,199)
150 PTH=1000:TZ=2000
160 '
170 GOSUB *DINP
180 COL=2:GOSUB *DISP
190 SX=1:SY=1.5:SZ=1.5:GOSUB *ENL
200 COL=4:GOSUB *DISP
210 END
220 '
230 *DINP
240 FOR I=1 TO N
250 READ P(I,0),P(I,1),P(I,2),P(I,3)
260 NEXT I
270 RETURN
280 '
290 *DISP
300 FOR I=1 TO N
310 X=P(I,0):Y=P(I,1):Z=P(I,2):F=P(I,3)
320 IF F>1 THEN PSET(PTH*X/(TZ-Z),PTH*Y
    /(TZ-Z)),COL:GOTO 350
```

```
330 ZOL=TZ-ZOLD:ZZ=TZ-Z
340 LINE(PTH*XOLD/ZOL,PTH*YOLD/ZOL)-(PT
H*X/ZZ,PTH*Y/ZZ),COL
350 XOLD=X:YOLD=Y:ZOLD=Z
360 NEXT I
370 RETURN
380 '
390 *ENL
400 FOR I=1 TO N
410 P(I,0)=SX*P(I,0)
420 P(I,1)=SY*P(I,1)
430 P(I,2)=SZ*P(I,2)
440 NEXT I
450 RETURN
460 '
470 DATA 100,100,100,2
480 DATA 100,200,100,1
490 DATA 200,200,100,1
500 DATA 200,100,100,1
510 DATA 100,100,100,1
520 DATA 100,100,200,1
530 DATA 200,100,200,1
540 DATA 200,200,200,1
550 DATA 100,200,200,1
560 DATA 100,100,200,1
570 DATA 100,200,100,2
580 DATA 100,200,200,1
590 DATA 200,200,100,2
600 DATA 200,200,200,1
610 DATA 200,100,100,2
620 DATA 200,100,200,1
```

5・4 ワイヤーフレーム・モデルを作る

ここまで勉強したことがらを利用してワイヤーフレーム・モデルを作ってみましょう。テレビのCMなどで最近よく見かける針金細工のような物体のモデルのことをワイヤーフレーム・モデルといいます。一番良くみかけるのは建築物です。これは作りやすいのですが、あまり面白くない上に、本式にやるとなるといろいろなきまりがあって、やかましいので、宇宙船とかアンテナとかおもしろそうな物体を題材にしましょう。

最初はパラボラ・アンテナを題材にしてみました。ワイヤーフレームとは針金の枠組みぐらいの意味ですし、実際にパラボラ・アンテナのうちには鉄骨だけで作られているものもあるので、リアリティが増します。

プログラムはいくつかのサブルーチンに分けて作られています。

＊DINPはパラボラの回転断面のデータ（2050行から2200行の間で与えられています）を読み込みます。データは実在のアンテナからとりました。CRTで見ると断面形状がはっきりと見えるでしょう。

＊MKDATAは断面データを回転させてパラボラ・アンテナを作るものです。X軸のまわりに回転させています。最終データは配列 P（★, ★, ★）に格納します。

＊ROTATEはパラボラを３次元空間の任意の方向に回転させます。

Z軸方向の回転をバンク
Y軸方向の回転をヘッディング
X軸方向の回転をピッチ

と呼ぶのは、慣例を踏襲しました。回転の終ったデータは配列VP(★, ★, ★)に格納され、もとのデータは配列P（★, ★, ★）に保存されています。

＊DISP1は骨組を表示するものです。

＊DISP2はもう一組の骨組を表示するものです。

PRINT USINGにはバージョンによって、ROMにバグがあり、あるデータの場合には暴走することがあるようです。小数点と＃の組み合わせによってはエラーがでますので、注意して使って下さい。

●プログラム5－4●

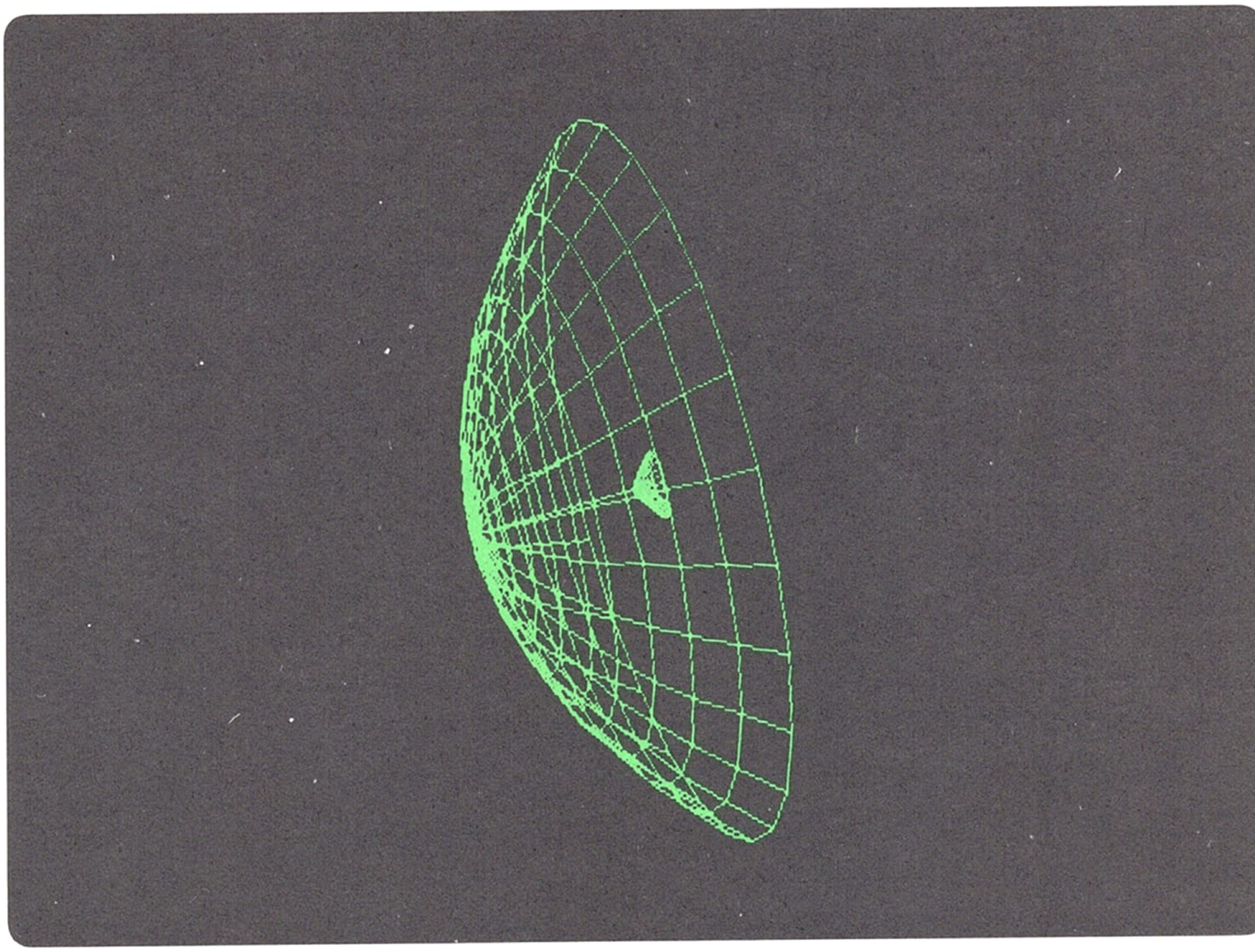
```
1000 *MAIN
1010 MAXN=16:DIV=25
1020 DIM D(MAXN,1),P(MAXN,DIV,2),VP(MAXN,DIV,2)
        ,DP(MAXN,2)
1030 CONSOLE 0,25,0,1:WIDTH 80,25
1040 SCREEN 0,3:CLS 2:SCREEN 0,0
1050 VIEW (0,0)-(639,199)
1060 WINDOW (-160,-100)-(160,100)
1070 TX=0:TY=0:TZ=1000:PTH=2000
1080 COL=4:PAI=3.14159
1090 GOSUB *DINP             ← 回転断面のデータ読み込み
1100 GOSUB *MKDATA           ← X軸のまわりに回転させる
1110 GOSUB *ROTATE           ← 3次元空間内の任意の方向に回転させる
1120 COL=6:GOSUB *DISP1      ← アンテナの骨組を作る
1130 COL=3:GOSUB *DISP2      ← 紫色の線でアンテナを仕上げる
1140 END
1150 '
1160 *DINP
1170 RESTORE *SAMPLE
1180 READ N
1190 FOR I=0 TO N
1200 READ X,Y
1210 PRINT USING "x=#### , y =####";X,Y  ← データを画面に表示
1220 D(I,0)=X:D(I,1)=Y
1230 NEXT I
1240 FOR I=0 TO N-1
1250 LINE(D(I,0),-D(I,1))-(D(I+1,0),-D(I+1,1)),4
                                   断面図を画面に表示 ↑
1260 NEXT I
1270 RETURN
1280 '
1290 *MKDATA
1300 CONSOLE 0,25
1310 FOR I=0 TO DIV-1
1320 RAD=PAI*2*I/DIV
1330 FOR J=0 TO N
1340 P(J,I,0)=D(J,0)
1350 P(J,I,1)=D(J,1)*COS(RAD)
1360 P(J,I,2)=-D(J,1)*SIN(RAD)
1370 NEXT J
1380 NEXT I
1390 '
1400 FOR J=0 TO N
```

```
1410 FOR K=0 TO 2
1420 P(J,DIV,K)=P(J,0,K)
1430 NEXT K
1440 NEXT J
1450 RETURN
1460 '
1470 *ROTATE
1480 CLS
1490 INPUT "Z rotation (Deg.) bank      =";B
     :B=PAI*B/180
1500 INPUT "Y rotation (Deg.) heading   =";H
     :H=PAI*H/180
1510 INPUT "X rotation (Deg.) pitch     =";P
     :P=PAI*P/180
1520 '
1530 FOR I=0 TO N
1540 FOR J=0 TO DIV
1550 X=P(I,J,0):Y=P(I,J,1):Z=P(I,J,2)
1560 GOSUB *MOVE3D
1570 VP(I,J,0)=X:VP(I,J,1)=Y:VP(I,J,2)=Z
1580 NEXT J
1590 NEXT I
1600 RETURN
1610 '
1620 *MOVE3D
1630 W=X*COS(B)-Y*SIN(B):Y=X*SIN(B)+Y*COS(B)
     :X=W
1640 W=X*COS(H)+Z*SIN(H):Z=-X*SIN(H)+Z*COS(H)
     :X=W
1650 W=Y*COS(P)-Z*SIN(P):Z=Y*SIN(P)+Z*COS(P)
     :Y=W
1660 RETURN
1670 '
1680 *DISP1
1690 SCREEN 0,3:CLS 3:SCREEN 0,0
1700 WINDOW (-320,-200)-(320,200)
1710 FOR I=0 TO DIV
1720 PCNT=0
1730 FOR J=0 TO N
1740 X=VP(J,I,0):Y=-VP(J,I,1):Z=VP(J,I,2)
1750 COLOR 5:LOCATE 0,0
1760 PRINT USING "X=####, Y=####, Z=####";X,Y,Z
1770 IF PCNT=0 THEN PSET(PTH*X/(TZ-Z+1),PTH*Y
```

X、Y、Z軸方向の回転角度入力

```
     /(TZ-Z+1)),COL:GOTO 1800
1780 ZOL=TZ-ZOLD+1:ZZ=TZ-Z+1
1790 LINE(PTH*XOLD/ZOL,PTH*YOLD/ZOL)-(PTH*X/
     ZZ,PTH*Y/ZZ),COL
1800 XOLD=X:YOLD=Y:ZOLD=Z
1810 PCNT=1
1820 NEXT J
1830 NEXT I
1840 RETURN
1850 '
1860 *DISP2
1870 CLS
1880 FOR I=0 TO N
1890 PCNT=0
1900 FOR J=0 TO DIV
1910 X=VP(I,J,0):Y=-VP(I,J,1):Z=VP(I,J,2)
1920 COLOR 5:LOCATE 0,0
1930 PRINT USING "X=####, Y=####, Z=####";X,Y,Z
1940 IF PCNT=0 THEN PSET(PTH*X/(TZ-Z+1),PTH*Y
     /(TZ-Z+1)),COL:GOTO 1970
1950 ZOL=TZ-ZOLD+1:ZZ=TZ-Z+1
1960 LINE(PTH*XOLD/ZOL,PTH*YOLD/ZOL)-(PTH*X/
     ZZ,PTH*Y/ZZ),COL
1970 XOLD=X:YOLD=Y:ZOLD=Z
1980 PCNT=1
1990 NEXT J
2000 NEXT I
2010 RETURN
2020 *SAMPLE
2030 DATA 15
2040 '
2050 DATA 45.7, 85.7
2060 DATA 36.9, 76.7
2070 DATA 29.0, 68.1
2080 DATA 22.4, 59.6
2090 DATA 16.4, 51.4
2100 DATA 12.8, 44.4
2110 DATA  8.3, 36.3
2120 DATA  5.1, 28.0
2130 DATA  2.5, 19.6
2140 DATA  0.8, 11.6
2150 DATA  0.0,  0.0
2160 DATA 39.3,  0.0
```

```
2170 DATA 39.6,  1.5
2180 DATA 41.3,  3.7
2190 DATA 43.2,  5.8
2200 DATA 44.9,  8.0
2210 '
```



プログラムをRUNさせると、YZ平面内の断面が表示され、これをX軸のまわりに回転させることになります。しばらく表示には変化がありませんが、パラボラの回転データを作っているのです。回転データができ上がると、空間のどの方向に回転させるかを聞いてきますので、回転角を入力して下さい。こんどはもう少し時間がかかりますが処理結果が図のように表示されてきます。

プログラムにいろいろ手を入れるともっと面白くなりますが、入力しやすいようにできる限り簡単にしてあります。

5·5 地球を作ってみよう

前のプログラムを利用すると地球のワイヤーフレーム・モデルを作ることができます。地球のように球体に近いものはディジタイザーで入力したり、方眼紙を使うよりも、式でデータを与えた方が楽です。そこで、今度はプログラム5-4の2050行から 2200行のデータを切り捨てて、次のように＊DINPサブルーチンを書き直してやります。

```
1160 *DINP:N=15
1170 FOR I=0 TO N
1180 X=50*COS(2*PAI*I/32):Y=50*SIN(2*PAI*I/32)
1190 PRINT X,Y
1200 D(I,0)=X:D(I,1)=Y
1210 NEXT I
1220 FOR I=0 TO N-1
1230 LINE(D(I,0),-D(I,1))-(D(I+1,0),-D(I+1,1)),4
1240 NEXT I
1250 RETURN
1260 '
```

プログラムをRUNさせると、前と同じように回転角を聞いてきますから、適当に入れて下さい。スーパーマンの眼で見た地球で、裏側まで見えてしまいます。どうしたら裏側が見えなくなるかは陰線消去の問題と言って大変おもしろい問題です。

地球を回すにはどうしたら良いでしょう。

❶ SCREENをきりかえる方法

❷ パレットをきりかえる方法

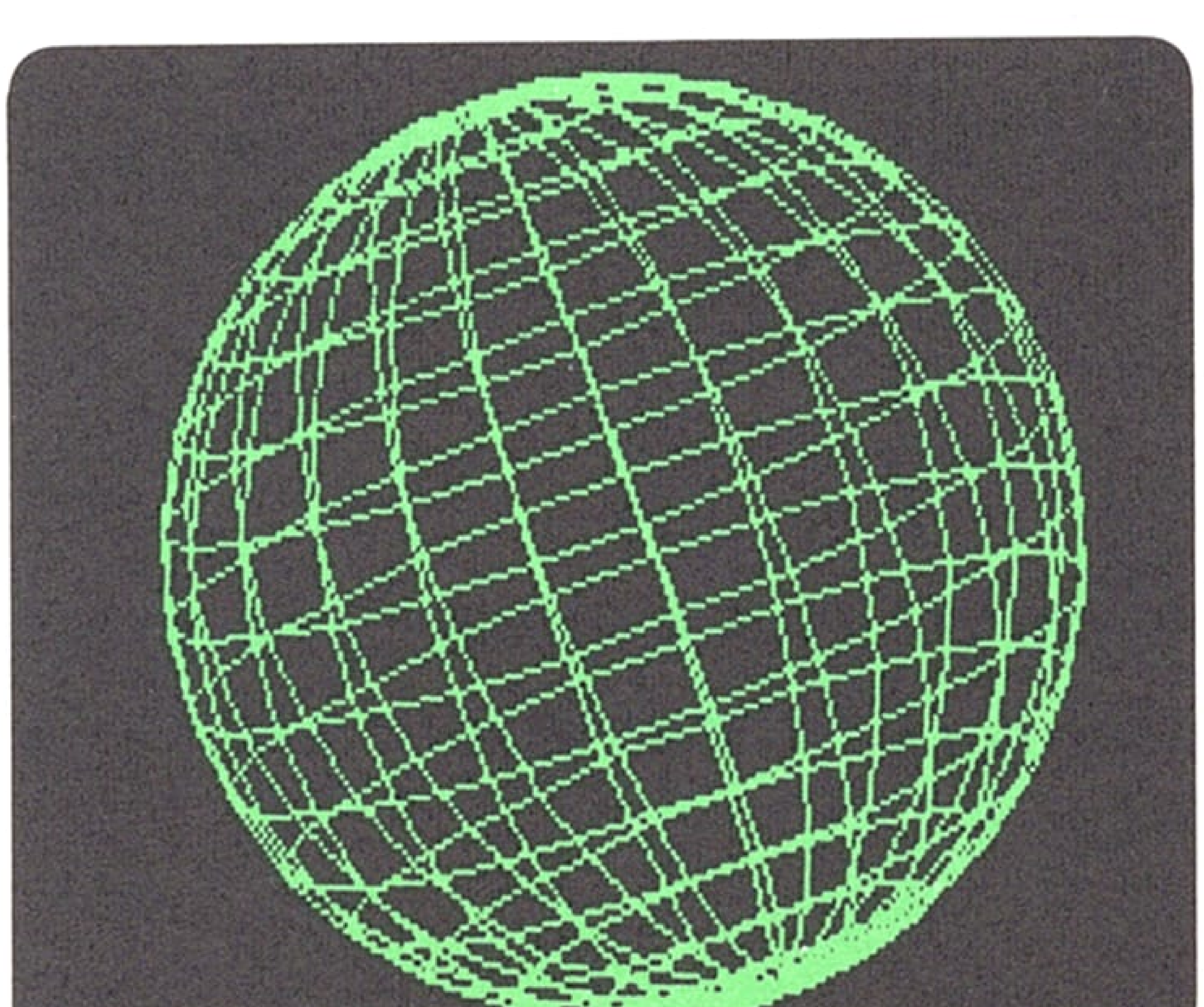
どちらにしても回っているように見せるには３画面は必要です。これまで勉強されて来られた皆さんには、もうおわかりですね。いろいろ試みて下さい。

5·6 地球に穴をあける

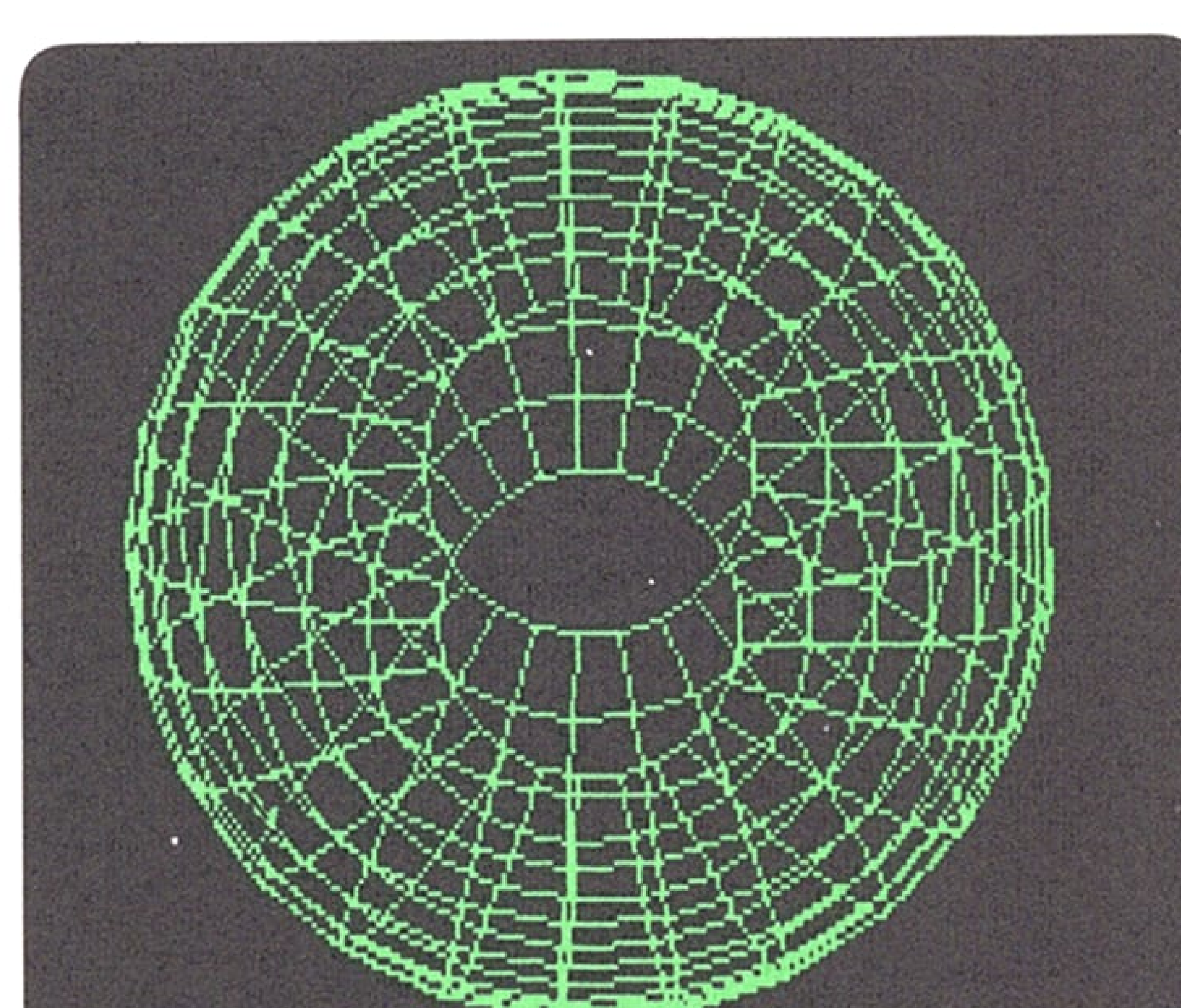
少しぶっそうな表題ですが、データの与え方をかえると、地球に穴をあけることができます。簡単なテクニックですが、結果の図がおもしろく採録しておくことにしました。

```
1160 *DINP:N=14
1170 FOR I=0 TO N
1180 X=50*COS(2*PAI*(I+2)/36)
    :Y=50*SIN(2*PAI*(I+2)/36)
1190 PRINT X,Y
1200 D(I,0)=X:D(I,1)=Y
1210 NEXT I
1220 FOR I=0 TO N-1
1230 LINE(D(I,0),-D(I,1))-(D(I+1,0),-D(I+1,1)),4
1240 NEXT I
1250 RETURN
```

今はあまり見かけませんが、包装用の紙ひものボールのようでもあります。つぼを作ったり皿を作ったりするのも簡単にできます。いろいろ試して見て下さい。データをひろう時は変化の少ない所はまばらに、変化の大きい所は細かくする方がPC-８８０１に負担をかけず、また高速処理をすることができます。データをあまり細かくとりすぎると時間ばかりかかって望ましくありません。



地球を作る



地球に穴をあける

5・7　宇宙船エンタープライズ号を作る

さて、いよいよ宇宙船エンタープライズ号を作り始めましょう。宇宙船エンタープライズ号は４つの部分からできています。

1　艦橋（ブリッジ）部分
2　船体主要部分
3　推進機部分　２基

これらを部分品に分けて作っていくことにしましょう。まず艦橋からとりかかることにします。艦橋だけは作り方が他の部分と異なります。図に示すように、まずＸ軸のまわりに回転させてから、Ｚ軸のまわりに90°回転させて引き起して下さい。引き起した形を見ると、それらしくできています。この場合データの総数が、

25×18＝450

と多くなります。相当の分量になりますが、熱心な読者のために５-10節に掲載いたしますが、プログラムが正しく動きさえすれば、データは簡単に引き出すこともできます。たとえば行番号1560と1570の間に次のようなルーチンを入れて下さい。

●プリンタの場合

```
LPRINT X,Y,Z
```

とする。

●フロッピー・ディスクの場合

プログラムのどこかで、

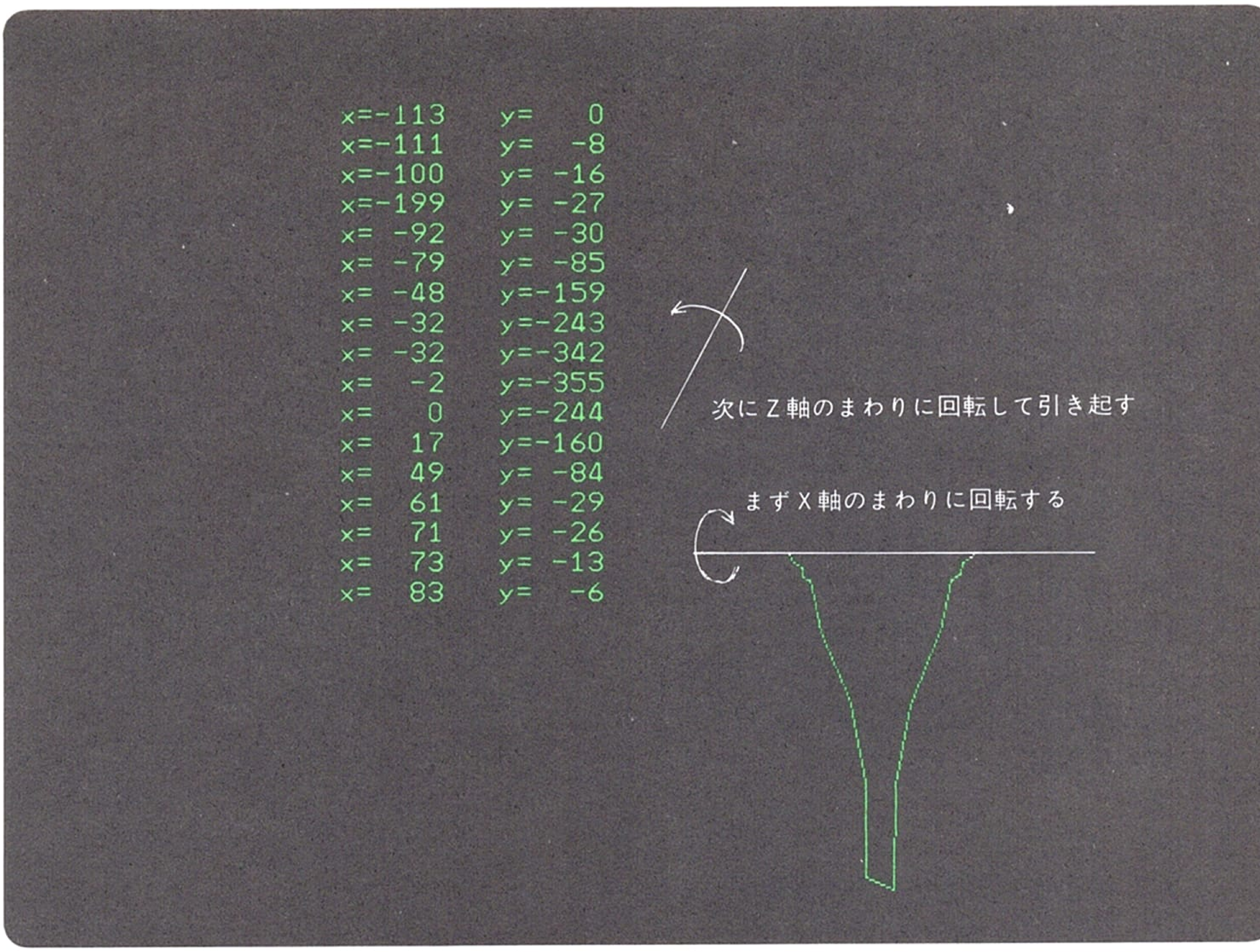
```
OPEN "STARBRIG" FOR OUTPUT AS #1
```

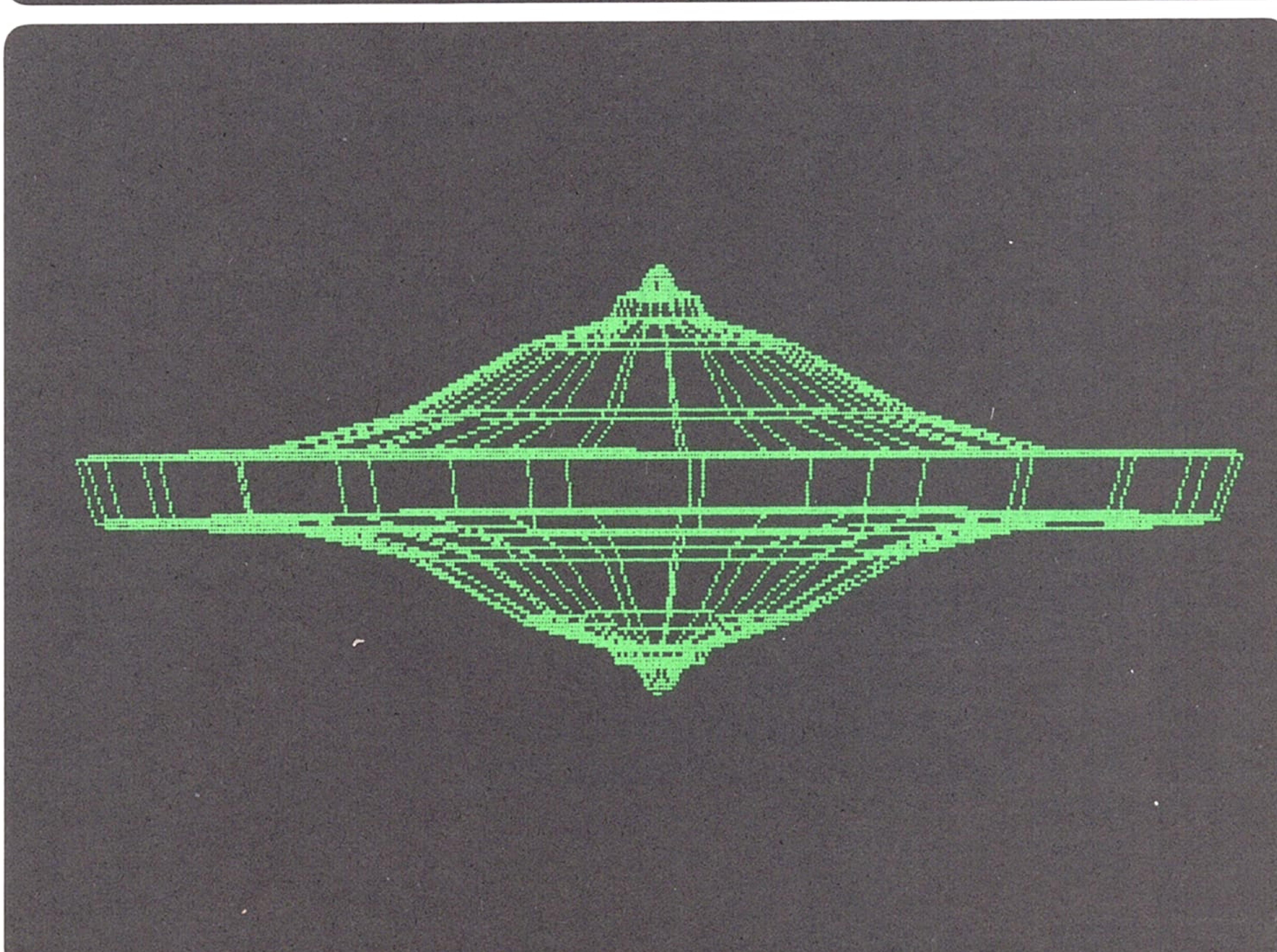
などとしておき、

```
PRINT #1,X,Y,Z
```

とします。

この場合、できればDATAの中にFの欄を作っておくと、線のひき始めがわかって便利です。PC-８８０１の能力を最大限に活用して、なるべく楽をしてやって下さい。


x=-113 y= 0
x=-111 y= -8
x=-100 y= -16
x=-199 y= -27
x= -92 y= -30
x= -79 y= -85
x= -48 y=-159
x= -32 y=-243
x= -32 y=-342
x= -2 y=-355
x= 0 y=-244
x= 17 y=-160
x= 49 y= -84
x= 61 y= -29
x= 71 y= -26
x= 73 y= -13
x= 83 y= -6
次にZ軸のまわりに回転して引き起す
まずX軸のまわりに回転する



●プログラム5－5●

```
1000 *MAIN
1010 MAXN=16:DIV=25
1020 DIM D(MAXN,1),P(MAXN,DIV,2),VP(MAXN,DIV,2)
         ,DP(MAXN,2)
1030 CONSOLE 0,25,0,1:WIDTH 80,25
1040 SCREEN 0,3:CLS 2:SCREEN 0,0
1050 VIEW (0,0)-(639,199)
1060 WINDOW (-640,-400)-(640,400)
1070 TX=0:TY=0:TZ=2000:PTH=2000
1080 COL=4:PAI=3.14159
1090 GOSUB *DINP
1100 GOSUB *MKDATA
1110 GOSUB *ROTATE
1120 COL=6:GOSUB *DISP1
1130 COL=3:GOSUB *DISP2
1140 END
1150 '
1160 *DINP
1170 RESTORE *SAMPLE
1180 READ N
1190 FOR I=0 TO N
1200 READ X,Y
1210 PRINT USING "x=#### , y =####";X,Y
1220 D(I,0)=X:D(I,1)=Y
1230 NEXT I
1240 FOR I=0 TO N-1
1250 LINE(D(I,0),-D(I,1))-(D(I+1,0),-D(I+1,1)),4
1260 NEXT I
1270 RETURN
1280 '
1290 *MKDATA
1300 CONSOLE 0,25
1310 FOR I=0 TO DIV-1
1320 RAD=PAI*2*I/DIV
1330 FOR J=0 TO N
1340 P(J,I,0)=D(J,0)
1350 P(J,I,1)=D(J,1)*COS(RAD)
1360 P(J,I,2)=-D(J,1)*SIN(RAD)
1370 NEXT J
1380 NEXT I
1390 '
1400 FOR J=0 TO N
```

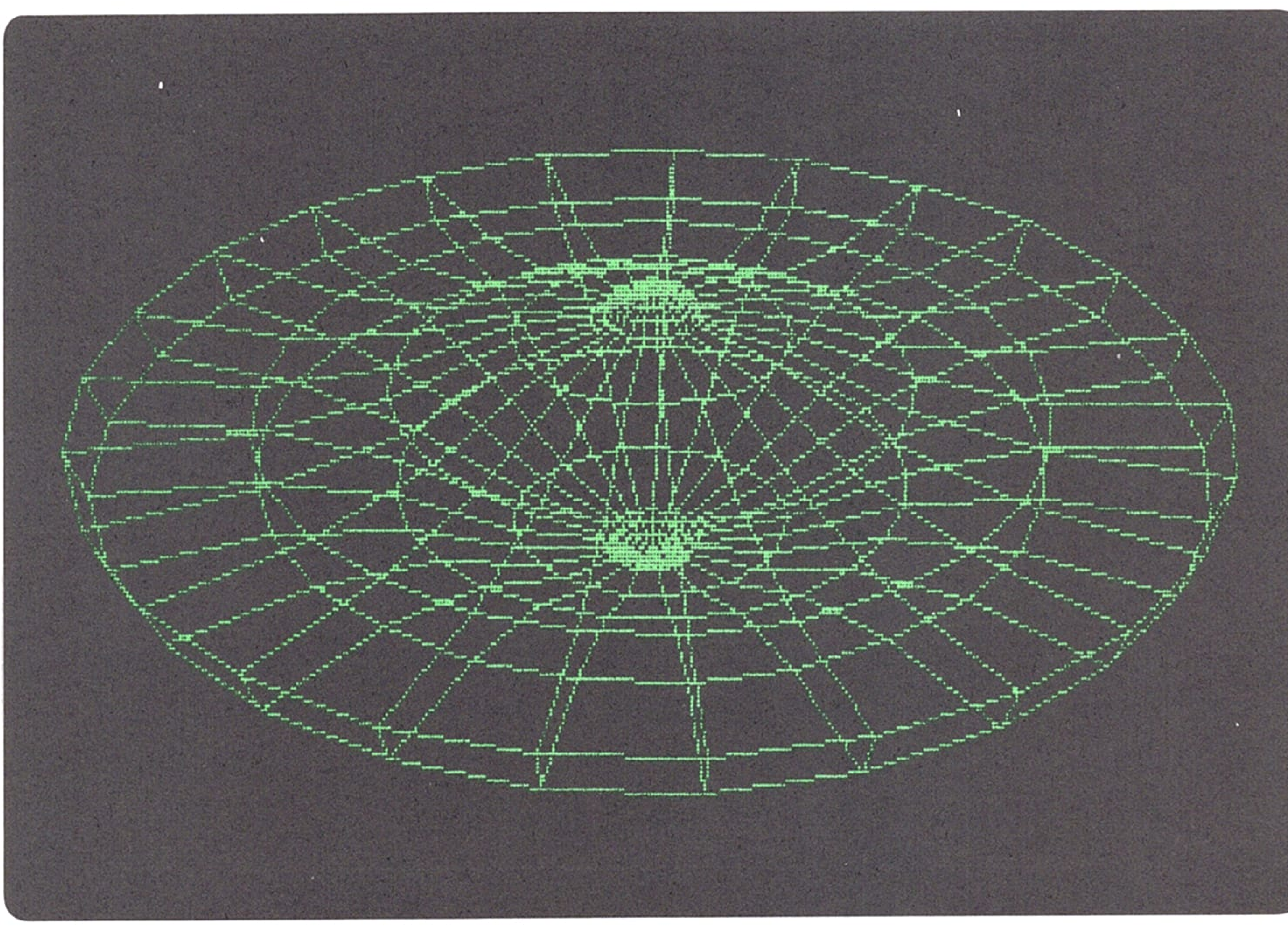
●このプログラムは、プログラム5―4の1060、1070、1700行を修正し、あらたに2050行以降のデータ文をキーインしなおしたものです。

```
1410 FOR K=0 TO 2
1420 P(J,DIV,K)=P(J,0,K)
1430 NEXT K
1440 NEXT J
1450 RETURN
1460 '
1470 *ROTATE
1480 CLS
1490 INPUT "Z rotation (Deg.) bank      =";B
     :B=PAI*B/180
1500 INPUT "Y rotation (Deg.) heading   =";H
     :H=PAI*H/180
1510 INPUT "X rotation (Deg.) pitch     =";P
     :P=PAI*P/180
1520 '
1530 FOR I=0 TO N
1540 FOR J=0 TO DIV
1550 X=P(I,J,0):Y=P(I,J,1):Z=P(I,J,2)
1560 GOSUB *MOVE3D
1570 VP(I,J,0)=X:VP(I,J,1)=Y:VP(I,J,2)=Z
1580 NEXT J
1590 NEXT I
1600 RETURN
1610 '
1620 *MOVE3D
1630 W=X*COS(B)-Y*SIN(B):Y=X*SIN(B)+Y*COS(B)
     :X=W
1640 W=X*COS(H)+Z*SIN(H):Z=-X*SIN(H)+Z*COS(H)
     :X=W
1650 W=Y*COS(P)-Z*SIN(P):Z=Y*SIN(P)+Z*COS(P)
     :Y=W
1660 RETURN
1670 '
1680 *DISP1
1690 SCREEN 0,3:CLS 3:SCREEN 0,0
1700 WINDOW (-640,-400)-(640,400)
1710 FOR I=0 TO DIV
1720 PCNT=0
1730 FOR J=0 TO N
1740 X=VP(J,I,0):Y=-VP(J,I,1):Z=VP(J,I,2)
1750 COLOR 5:LOCATE 0,0
1760 PRINT USING "X=####, Y=####, Z=####";X,Y,Z
1770 IF PCNT=0 THEN PSET(PTH*X/(TZ-Z+1)
```

```
     ,PTH*Y/(TZ-Z+1)),COL:GOTO 1800
1780 ZOL=TZ-ZOLD+1:ZZ=TZ-Z+1
1790 LINE(PTH*XOLD/ZOL,PTH*YOLD/ZOL)
         -(PTH*X/ZZ,PTH*Y/ZZ),COL
1800 XOLD=X:YOLD=Y:ZOLD=Z
1810 PCNT=1
1820 NEXT J
1830 NEXT I
1840 RETURN
1850 '
1860 *DISP2
1870 CLS
1880 FOR I=0 TO N
1890 PCNT=0
1900 FOR J=0 TO DIV
1910 X=VP(I,J,0):Y=-VP(I,J,1):Z=VP(I,J,2)
1920 COLOR 5:LOCATE 0,0
1930 PRINT USING "X=####, Y=####, Z=####";X,Y,Z
1940 IF PCNT=0 THEN PSET(PTH*X/(TZ-Z+1)
     ,PTH*Y/(TZ-Z+1)),COL:GOTO 1970
1950 ZOL=TZ-ZOLD+1:ZZ=TZ-Z+1
1960 LINE(PTH*XOLD/ZOL,PTH*YOLD/ZOL)
     -(PTH*X/ZZ,PTH*Y/ZZ),COL
1970 XOLD=X:YOLD=Y:ZOLD=Z
1980 PCNT=1
1990 NEXT J
2000 NEXT I
2010 RETURN
2020 *SAMPLE
2030 DATA 16
2040 '
2050 DATA -113,    0
2060 DATA -111,   -8
2070 DATA -100,  -16
2080 DATA  -99,  -27
2090 DATA  -92,  -30
2100 DATA  -79,  -85
2110 DATA  -48, -158
2120 DATA  -32, -243
2130 DATA  -32, -342
2140 DATA   -2, -355
2150 DATA    0, -244
2160 DATA   17, -160
```

```
2170 DATA   49,  -84
2180 DATA   61,  -29
2190 DATA   71,  -26
2200 DATA   73,  -13
2210 DATA   83,   -6
```

このプログラムでは艦橋部分をあらゆる角度から見ることができます。番傘かタイコ焼きのお化けのようでもありますが、構造がよくわかります。プリンタにおとすと細かい部分がつぶれてしまいますが、わりに細かく作られているのです。



5・8 推進機を作る

行番号2020からのDATAを次のように書き直して下さい。

```
2020 *SAMPLE
2030 DATA 11
2040 '
2050 DATA -363,   0
2060 DATA -361,  16
2070 DATA -341,  37
2080 DATA -324,  43
2090 DATA -319,  49
2100 DATA -300,  50
2110 DATA -296,  46
2120 DATA    0,  39
2130 DATA  388,  37
2140 DATA  403,  21
2150 DATA  409,  10
2160 DATA  410,  10
```

エンタープライズ号のマニュアルを読みますと、推進機はロケットエンジンのような平凡なものではなく、変わった説明が書いてあります。船体の重量も速度も我々の知らないような単位で書いてあります。あまりやかましいことさえ言わなければ簡単に作れます。

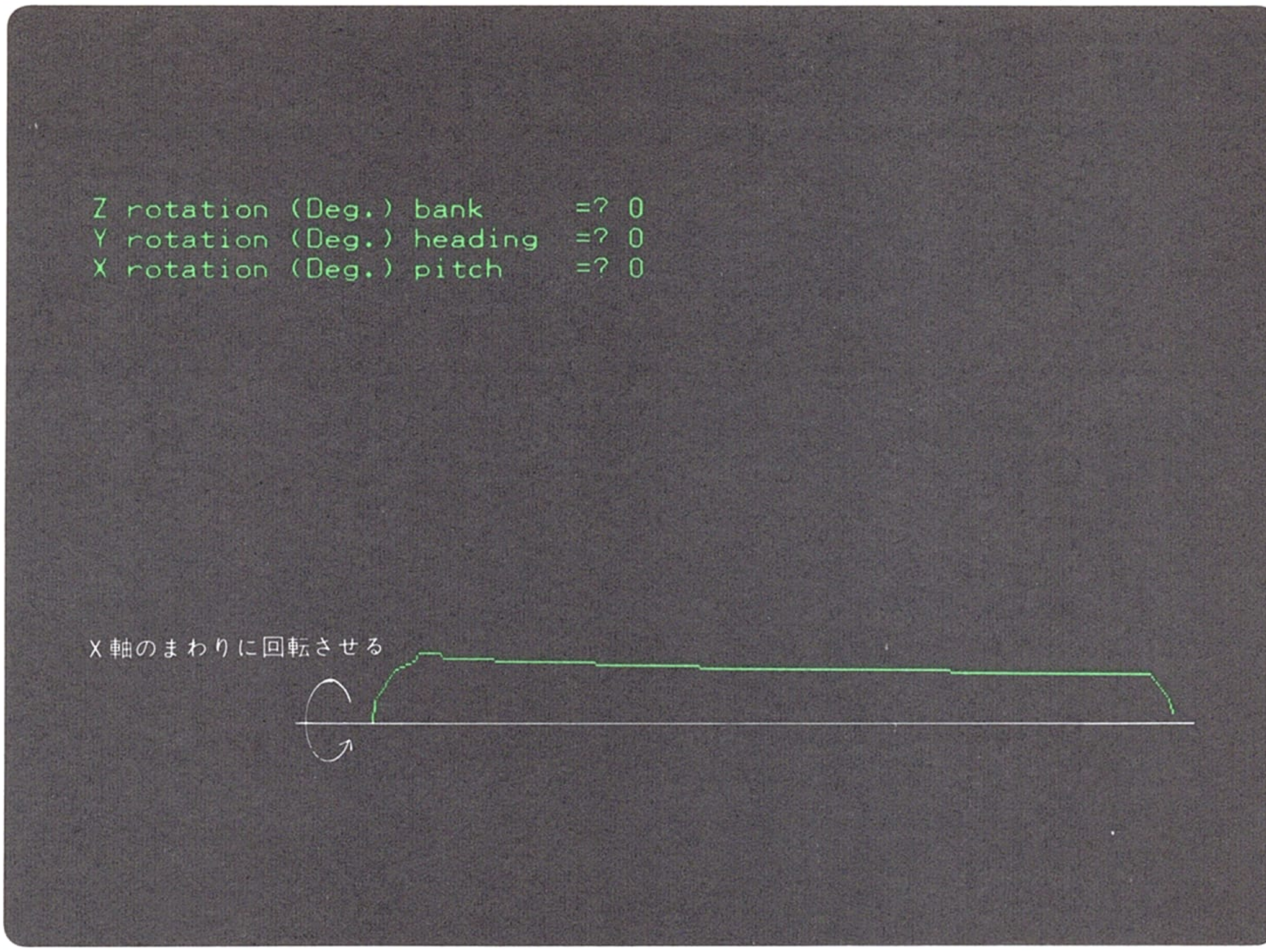
推進機の断面をx軸のまわりに回転させれば、それででき上がりです。難しいのはデータを拾う方の作業です。良い図面や写真は大変貴重なものです。日頃から気をつけて集めておくと役に立ちます。

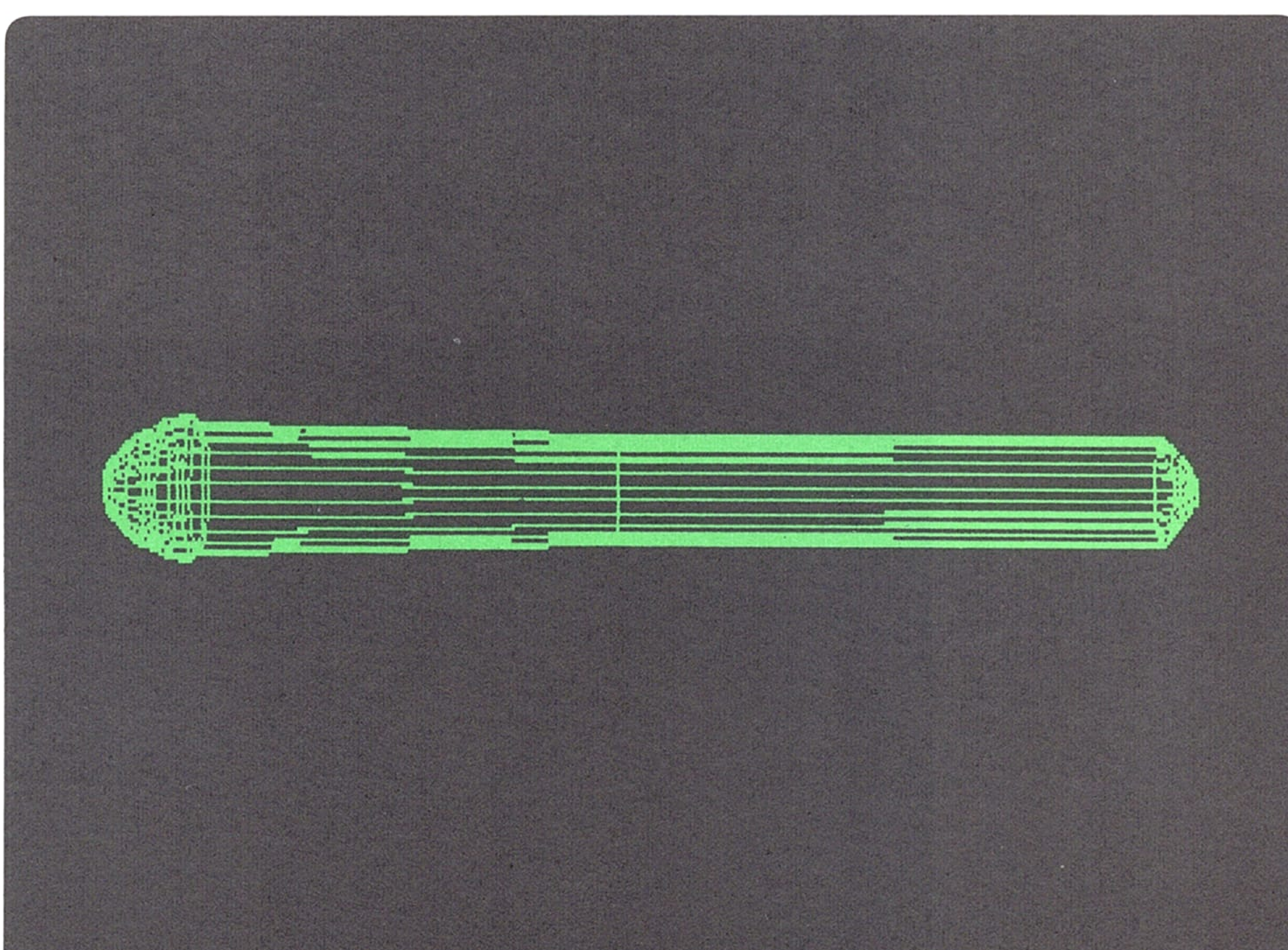
３Ｄ表示も真横とか真正面からよりも、ある程度角度をつけてやりますとずっと、それらしく見えます。

推進機のデータも、

25×12＝300

個になります。５－10節にデータを掲載します。


Z rotation (Deg.) bank =? 0
Y rotation (Deg.) heading =? 0
X rotation (Deg.) pitch =? 0
X軸のまわりに回転させる



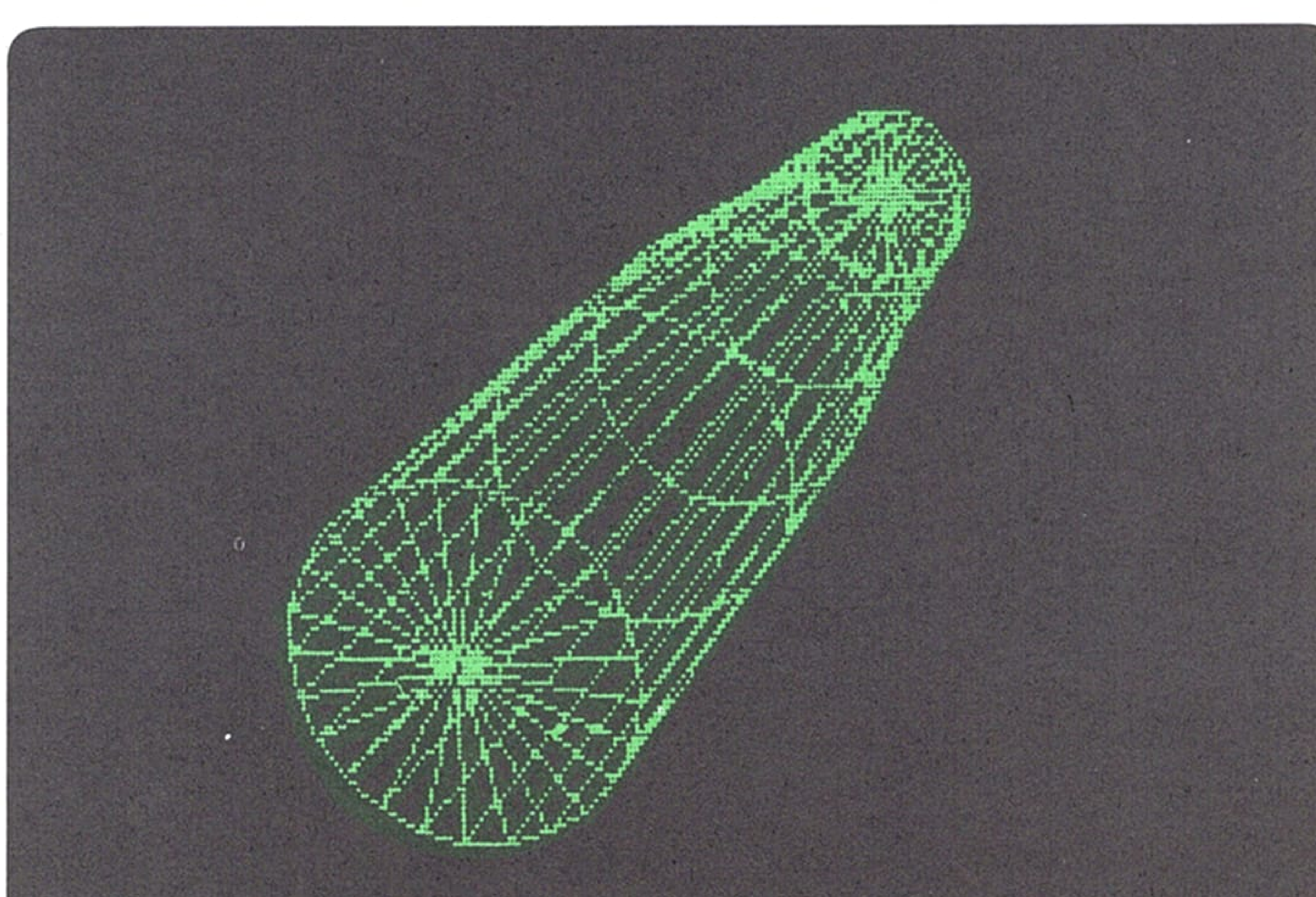
5・9　船体主要部分を作る

エンタープライズ号の船体主要部分は本当は回転対称ではないのでこれを回転体で近似することには若干問題があります。しかし、あまりこるのも大変ですから、回転体で我慢することにしましょう。

プログラムの一部を次のように書き直して下さい。

```
2020 *SAMPLE
2030 DATA 8
2040 '
2050 DATA -251,   3
2060 DATA -248,  74
2070 DATA  -33,  69
2080 DATA   90,  65
2090 DATA  212,  51
2100 DATA  322,  44
2110 DATA  341,  41
2120 DATA  352,  43
2130 DATA  351,   0
```

図面では船体主要部の前方にレーダー・アンテナがついています。本格派の方は前の方の節のパラボラ・アンテナのデータを利用してとりつけて下さい。ただし、この型のアンテナは副反射鏡がブロッキングするので、21世紀にはオフセット・パラボラ・アンテナにおきかわっているだろうと思います（現在はレーダー用としてはあまり使われていません。衛星通信用が多いようです）。いろいろ試して下さい。



5・10 エンタープライズ号の完成

部分品にわけて作ってきたエンタープライズ号を合体させ完成しましょう。データはかなり長いのですが、一応完全なデータを示してあります。私はフロッピーディスクでデータを管理していますが、フロッピーディスクをお持ちでない方のために、できる限り短いメインプログラムで処理できるように工夫してみました。

このプログラムの後ろに186ページからのデータをつけ加えてください。行番号は何番からでもかまいませんが、たとえば次のような形にして、データの最後には必ず0, 0, 0, 0をつけておいて下さい。これが終了の印となります。

```
10000 DATA -330,  22,  -1, 2,-338,  24,  -1, 1
10010 DATA -346,  35,  -1, 1,-357,  36,  -1, 1
10020 '
11000 DATA    0,   0,   0, 0
```

●プログラム5－6●

```
100 WIDTH 80,25:CONSOLE 0,25,0,1
110 SCREEN 0,3:CLS 3:SCREEN 0,0
120 WINDOW (-640,-400)-(640,400)
    :VIEW (0,0)-(639,199)
130 PTH=1000:TZ=2000
140 COL=7:PAI=3.14159
150 I=0
160 '
170 CLS:COLOR 5
180 INPUT "Z rotation (Deg.) bank       =
";B:B=PAI*B/180
190 INPUT "Y rotation (Deg.) heading    =
";H:H=PAI*H/180
200 INPUT "X rotation (Deg.) pitch      =
";P:P=PAI*P/180
210 '
220 RESTORE 10000
230 READ X,Y,Z,F
240 IF F=0 THEN END
250 '
260 W=X*COS(B)-Y*SIN(B):Y=X*SIN(B)+Y*COS
(B):X=W
270 W=X*COS(H)+Z*SIN(H):Z=-X*SIN(H)+Z*CO
```

```
S(H):X=W
280 W=Y*COS(P)-Z*SIN(P):Z=Y*SIN(P)+Z*COS
(P):Y=W
290 '
300 *DISP
310 Y=-Y
320 IF F>1 THEN PSET(PTH*X/(TZ-Z+1),PTH*
Y/(TZ-Z+1)),COL:GOTO 350
330 ZOL=TZ-ZOLD+1:ZZ=TZ-Z+1
340 LINE(PTH*XOLD/ZOL,PTH*YOLD/ZOL)-(PTH
*X/ZZ,PTH*Y/ZZ),COL
350 XOLD=X:YOLD=Y:ZOLD=Z
360 '
370 I=I+1
380 GOTO 230
```

ハードコピー用のルーチンをつけるとか、次々に回転をくり返して行くようにとか、いくらでもアイデアが出せると思います。この本で展開してきたテクニックを生かせば、かなり面白いことができると思います。皆さんの創造性を発揮してみて下さい。

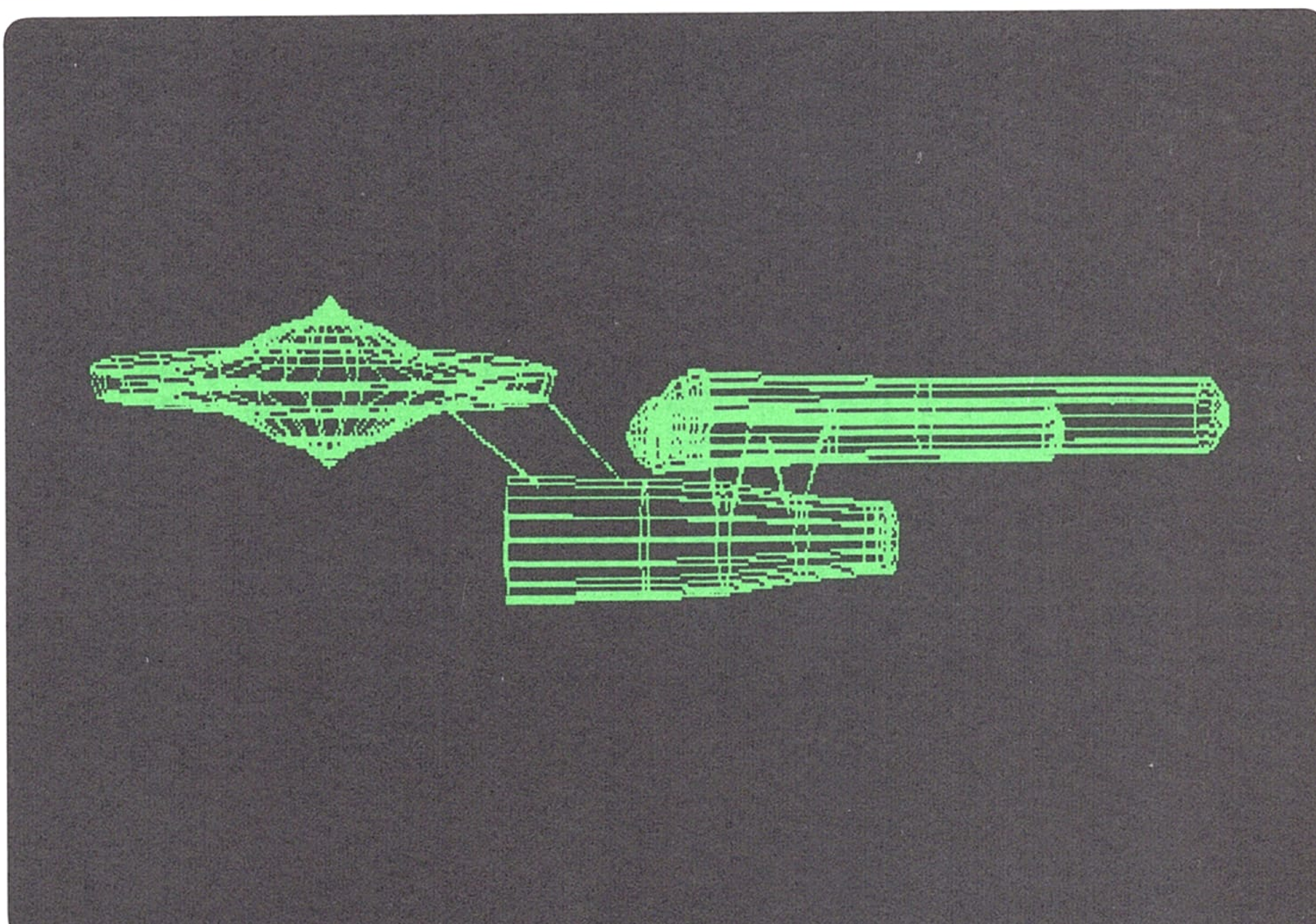
入力確認用プログラム

データ文が長いので入力確認用プログラムを利用して下さい。データ文がキチンと入っていれば走らせると、右ページの図のように真横から見た図がCRT上で見れるはずです。

●プログラム5－7●

```
100 WIDTH 80,25:CONSOLE 0,25,0,1
110 SCREEN 0,3:CLS 3:SCREEN 0,0
120 WINDOW (-640,-400)-(640,400)
130 VIEW (0,0)-(639,199)
140 PTH=1000:TZ=2000
150 COL=7:GOSUB *DINP
160 END
170 '
180 *DINP
190 READ X,Y,Z,F
200 IF F=0 THEN RETURN
210 GOSUB *DISP
220 GOTO 190
```

```
230 '
240 *DISP
250 Y=-Y
260 IF F>1 THEN PSET(PTH*X/(TZ-Z),PTH*Y/
(TZ-Z)),COL:GOTO 290
270 ZOL=TZ-ZOLD:ZZ=TZ-Z
280 LINE(PTH*XOLD/ZOL,PTH*YOLD/ZOL)-(PTH
*X/ZZ,PTH*Y/ZZ),COL
290 XOLD=X:YOLD=Y:ZOLD=Z
300 RETURN
```



エンタープライズ号作成データ

```
-330 , 22 ,-1 , 2
-338 , 24 ,-1 , 1
-346 , 35 ,-1 , 1
-357 , 36 ,-1 , 1
-360 , 43 ,-1 , 1
-415 , 56 ,-1 , 1
-488 , 87 ,-1 , 1
-573 , 103 ,-1 , 1
-672 , 102 ,-1 , 1
-685 , 133 ,-1 , 1
-574 , 135 ,-1 , 1
-490 , 152 ,-1 , 1
-414 , 184 ,-1 , 1
-359 , 196 ,-1 , 1
-356 , 206 ,-1 , 1
-343 , 208 ,-1 , 1
-336 , 218 ,-1 , 1
-329 , 221 ,-1 , 1
-330 , 22 ,-1 , 2
-338 , 24 ,-4 , 1
-345 , 35 ,-6 , 1
-356 , 36 ,-8 , 1
-359 , 43 ,-9 , 1
-412 , 56 ,-23 , 1
-483 , 87 ,-42 , 1
-565 , 103 ,-64 , 1
-660 , 102 ,-90 , 1
-673 , 133 ,-93 , 1
-566 , 135 ,-65 , 1
-484 , 152 ,-43 , 1
-411 , 184 ,-23 , 1
-358 , 196 ,-9 , 1
-355 , 206 ,-8 , 1
-342 , 208 ,-5 , 1
-336 , 218 ,-3 , 1
-329 , 221 ,-1 , 1
-330 , 22 ,-1 , 2
-337 , 24 ,-5 , 1
-344 , 35 ,-9 , 1
-353 , 36 ,-15 , 1
-356 , 43 ,-16 , 1
-404 , 56 ,-44 , 1
-467 , 87 ,-80 , 1
-540 , 103 ,-123 , 1
-626 , 102 ,-172 , 1
-637 , 133 ,-179 , 1
-541 , 135 ,-123 , 1
-468 , 152 ,-81 , 1
-403 , 184 ,-43 , 1
-355 , 196 ,-16 , 1
-352 , 206 ,-14 , 1
-341 , 208 ,-8 , 1
-335 , 218 ,-4 , 1
-329 , 221 ,-1 , 1
-330 , 22 ,-1 , 2
-336 , 24 ,-7 , 1
-341 , 35 ,-13 , 1
-349 , 36 ,-21 , 1
-351 , 43 ,-23 , 1
-390 , 56 ,-62 , 1
-442 , 87 ,-113 , 1
-502 , 103 ,-173 , 1
-572 , 102 ,-243 , 1
-581 , 133 ,-253 , 1
-502 , 135 ,-174 , 1
-443 , 152 ,-115 , 1
-389 , 184 ,-61 , 1
-350 , 196 ,-22 , 1
-348 , 206 ,-20 , 1
-339 , 208 ,-11 , 1
-334 , 218 ,-6 , 1
-329 , 221 ,-1 , 1
-330 , 22 ,-1 , 2
-334 , 24 ,-8 , 1
-338 , 35 ,-15 , 1
-343 , 36 ,-25 , 1
-345 , 43 ,-27 , 1
-372 , 56 ,-75 , 1
-409 , 87 ,-138 , 1
-451 , 103 ,-212 , 1
-501 , 102 ,-298 , 1
-507 , 133 ,-309 , 1
-452 , 135 ,-213 , 1
-410 , 152 ,-140 , 1
-372 , 184 ,-74 , 1
-344 , 196 ,-27 , 1
-343 , 206 ,-24 , 1
-336 , 208 ,-13 , 1
-333 , 218 ,-7 , 1
-329 , 221 ,-1 , 1
-330 , 22 ,-1 , 2
-332 , 24 ,-9 , 1
-334 , 35 ,-17 , 1
-337 , 36 ,-28 , 1
-338 , 43 ,-30 , 1
-352 , 56 ,-84 , 1
-371 , 87 ,-154 , 1
-393 , 103 ,-236 , 1
-418 , 102 ,-332 , 1
-422 , 133 ,-344 , 1
-393 , 135 ,-237 , 1
-371 , 152 ,-156 , 1
-352 , 184 ,-83 , 1
-337 , 196 ,-30 , 1
-337 , 206 ,-27 , 1
-333 , 208 ,-14 , 1
-331 , 218 ,-7 , 1
-330 , 221 ,-1 , 1
-330 , 22 ,-1 , 2
-330 , 24 ,-9 , 1
-330 , 35 ,-17 , 1
-330 , 36 ,-28 , 1
-330 , 43 ,-31 , 1
-330 , 56 ,-86 , 1
-330 , 87 ,-159 , 1
-330 , 103 ,-244 , 1
-330 , 102 ,-343 , 1
-330 , 133 ,-356 , 1
-330 , 135 ,-245 , 1
-330 , 152 ,-161 , 1
-330 , 184 ,-85 , 1
-330 , 196 ,-30 , 1
-330 , 206 ,-27 , 1
-330 , 208 ,-14 , 1
-330 , 218 ,-7 , 1
-330 , 221 , 0 , 1
-330 , 22 ,-1 , 2
-328 , 24 ,-9 , 1
-326 , 35 ,-17 , 1
-323 , 36 ,-28 , 1
-322 , 43 ,-30 , 1
-308 , 56 ,-84 , 1
-289 , 87 ,-154 , 1
-267 , 103 ,-236 , 1
-241 , 102 ,-332 , 1
-238 , 133 ,-344 , 1
-267 , 135 ,-237 , 1
-289 , 152 ,-156 , 1
-308 , 184 ,-83 , 1
-322 , 196 ,-30 , 1
-323 , 206 ,-27 , 1
-327 , 208 ,-14 , 1
-328 , 218 ,-7 , 1
-330 , 221 ,-1 , 1
-330 , 22 ,-1 , 2
-326 , 24 ,-8 , 1
-322 , 35 ,-15 , 1
-316 , 36 ,-25 , 1
-315 , 43 ,-27 , 1
-287 , 56 ,-75 , 1
-251 , 87 ,-138 , 1
-208 , 103 ,-212 , 1
-159 , 102 ,-298 , 1
-152 , 133 ,-309 , 1
-208 , 135 ,-213 , 1
-250 , 152 ,-140 , 1
```

```
-288 , 184 ,-74 , 1
-315 , 196 ,-27 , 1
-317 , 206 ,-24 , 1
-323 , 208 ,-13 , 1
-327 , 218 ,-7 , 1
-330 , 221 ,-1 , 1
-330 , 22 ,-1 , 2
-324 , 24 ,-7 , 1
-319 , 35 ,-13 , 1
-311 , 36 ,-21 , 1
-309 , 43 ,-23 , 1
-270 , 56 ,-62 , 1
-218 , 87 ,-113 , 1
-158 , 103 ,-173 , 1
-88 , 102 ,-243 , 1
-79 , 133 ,-253 , 1
-157 , 135 ,-174 , 1
-217 , 152 ,-115 , 1
-271 , 184 ,-61 , 1
-309 , 196 ,-22 , 1
-312 , 206 ,-20 , 1
-321 , 208 ,-11 , 1
-326 , 218 ,-6 , 1
-331 , 221 ,-1 , 1
-330 , 22 ,-1 , 2
-323 , 24 ,-5 , 1
-316 , 35 ,-9 , 1
-307 , 36 ,-15 , 1
-304 , 43 ,-16 , 1
-256 , 56 ,-44 , 1
-193 , 87 ,-81 , 1
-119 , 103 ,-123 , 1
-34 , 102 ,-173 , 1
-22 , 133 ,-179 , 1
-119 , 135 ,-123 , 1
-191 , 152 ,-82 , 1
-257 , 184 ,-44 , 1
-305 , 196 ,-16 , 1
-307 , 206 ,-14 , 1
-319 , 208 ,-8 , 1
-325 , 218 ,-4 , 1
-331 , 221 ,-1 , 1
-330 , 22 ,-1 , 2
-322 , 24 ,-4 , 1
-314 , 35 ,-6 , 1
-304 , 36 ,-8 , 1
-301 , 43 ,-9 , 1
-248 , 56 ,-23 , 1
-177 , 87 ,-42 , 1
-95 , 103 ,-64 , 1
 0 , 102 ,-90 , 1
 13 , 133 ,-93 , 1
-94 , 135 ,-65 , 1
-175 , 152 ,-43 , 1
```

```
-249 , 184 ,-23 , 1
-302 , 196 ,-9 , 1
-305 , 206 ,-8 , 1
-317 , 208 ,-5 , 1
-324 , 218 ,-3 , 1
-331 , 221 ,-1 , 1
-330 , 22 ,-1 , 2
-322 , 24 ,-1 , 1
-314 , 35 ,-1 , 1
-303 , 36 ,-2 , 1
-300 , 43 ,-2 , 1
-245 , 56 ,-2 , 1
-172 , 87 ,-2 , 1
-87 , 103 ,-2 , 1
 12 , 102 ,-2 , 1
 25 , 133 ,-2 , 1
-86 , 135 ,-2 , 1
-170 , 152 ,-2 , 1
-246 , 184 ,-2 , 1
-301 , 196 ,-2 , 1
-304 , 206 ,-2 , 1
-317 , 208 ,-1 , 1
-324 , 218 ,-1 , 1
-331 , 221 ,-1 , 1
-330 , 22 ,-1 , 2
-322 , 24 , 1 , 1
-314 , 35 , 3 , 1
-304 , 36 , 5 , 1
-301 , 43 , 6 , 1
-248 , 56 , 20 , 1
-177 , 87 , 39 , 1
-95 , 103 , 61 , 1
 0 , 102 , 87 , 1
 13 , 133 , 90 , 1
-94 , 135 , 62 , 1
-175 , 152 , 40 , 1
-249 , 184 , 20 , 1
-302 , 196 , 6 , 1
-305 , 206 , 5 , 1
-317 , 208 , 2 , 1
-324 , 218 , 0 , 1
-331 , 221 ,-2 , 1
-330 , 22 ,-1 , 2
-323 , 24 , 3 , 1
-316 , 35 , 7 , 1
-307 , 36 , 12 , 1
-304 , 43 , 13 , 1
-256 , 56 , 41 , 1
-193 , 87 , 77 , 1
-119 , 103 , 120 , 1
-34 , 102 , 169 , 1
-22 , 133 , 176 , 1
-119 , 135 , 121 , 1
-191 , 152 , 78 , 1
```

```
-257 , 184 , 40 , 1
-305 , 196 , 13 , 1
-307 , 206 , 11 , 1
-319 , 208 , 5 , 1
-325 , 218 , 2 , 1
-331 , 221 ,-2 , 1
-330 , 22 ,-1 , 2
-324 , 24 , 4 , 1
-319 , 35 , 10 , 1
-311 , 36 , 18 , 1
-309 , 43 , 20 , 1
-270 , 56 , 59 , 1
-218 , 87 , 110 , 1
-158 , 103 , 170 , 1
-88 , 102 , 240 , 1
-79 , 133 , 250 , 1
-157 , 135 , 171 , 1
-217 , 152 , 112 , 1
-271 , 184 , 58 , 1
-309 , 196 , 19 , 1
-312 , 206 , 17 , 1
-321 , 208 , 8 , 1
-326 , 218 , 3 , 1
-331 , 221 ,-2 , 1
-330 , 22 ,-1 , 2
-326 , 24 , 5 , 1
-322 , 35 , 12 , 1
-316 , 36 , 22 , 1
-315 , 43 , 24 , 1
-287 , 56 , 72 , 1
-251 , 87 , 135 , 1
-208 , 103 , 209 , 1
-159 , 102 , 295 , 1
-152 , 133 , 306 , 1
-208 , 135 , 210 , 1
-250 , 152 , 137 , 1
-288 , 184 , 71 , 1
-315 , 196 , 24 , 1
-317 , 206 , 21 , 1
-323 , 208 , 10 , 1
-327 , 218 , 4 , 1
-330 , 221 ,-2 , 1
-330 , 22 ,-1 , 2
-328 , 24 , 6 , 1
-326 , 35 , 14 , 1
-323 , 36 , 25 , 1
-322 , 43 , 27 , 1
-308 , 56 , 81 , 1
-289 , 87 , 151 , 1
-267 , 103 , 233 , 1
-241 , 102 , 329 , 1
-238 , 133 , 341 , 1
-267 , 135 , 234 , 1
-289 , 152 , 153 , 1
```

```
-308 , 184 , 80 , 1
-322 , 196 , 27 , 1
-323 , 206 , 24 , 1
-327 , 208 , 11 , 1
-328 , 218 , 4 , 1
-330 , 221 ,-2 , 1
-330 , 22 ,-1 , 2
-330 , 24 , 7 , 1
-330 , 35 , 15 , 1
-330 , 36 , 26 , 1
-330 , 43 , 29 , 1
-330 , 56 , 84 , 1
-330 , 87 , 157 , 1
-330 , 103 , 242 , 1
-330 , 102 , 341 , 1
-330 , 133 , 354 , 1
-330 , 135 , 243 , 1
-330 , 152 , 159 , 1
-330 , 184 , 83 , 1
-330 , 196 , 28 , 1
-330 , 206 , 25 , 1
-330 , 208 , 12 , 1
-330 , 218 , 5 , 1
-330 , 221 ,-2 , 1
-330 , 22 ,-1 , 2
-332 , 24 , 6 , 1
-334 , 35 , 14 , 1
-337 , 36 , 25 , 1
-338 , 43 , 27 , 1
-352 , 56 , 81 , 1
-371 , 87 , 151 , 1
-393 , 103 , 233 , 1
-418 , 102 , 329 , 1
-422 , 133 , 341 , 1
-393 , 135 , 234 , 1
-371 , 152 , 153 , 1
-352 , 184 , 80 , 1
-337 , 196 , 27 , 1
-337 , 206 , 24 , 1
-333 , 208 , 11 , 1
-331 , 218 , 4 , 1
-330 , 221 ,-2 , 1
-330 , 22 ,-1 , 2
-334 , 24 , 5 , 1
-338 , 35 , 12 , 1
-343 , 36 , 22 , 1
-345 , 43 , 24 , 1
-372 , 56 , 72 , 1
-409 , 87 , 135 , 1
-451 , 103 , 209 , 1
-501 , 102 , 295 , 1
-507 , 133 , 306 , 1
-452 , 135 , 210 , 1
-410 , 152 , 137 , 1
-372 , 184 , 71 , 1
-344 , 196 , 24 , 1
-343 , 206 , 21 , 1
-336 , 208 , 10 , 1
-333 , 218 , 4 , 1
-329 , 221 ,-2 , 1
-330 , 22 ,-1 , 2
-336 , 24 , 4 , 1
-341 , 35 , 10 , 1
-349 , 36 , 18 , 1
-351 , 43 , 20 , 1
-390 , 56 , 59 , 1
-442 , 87 , 110 , 1
-502 , 103 , 170 , 1
-572 , 102 , 240 , 1
-581 , 133 , 250 , 1
-502 , 135 , 171 , 1
-443 , 152 , 112 , 1
-389 , 184 , 58 , 1
-350 , 196 , 19 , 1
-348 , 206 , 17 , 1
-339 , 208 , 8 , 1
-334 , 218 , 3 , 1
-329 , 221 ,-2 , 1
-330 , 22 ,-1 , 2
-337 , 24 , 3 , 1
-344 , 35 , 7 , 1
-353 , 36 , 12 , 1
-356 , 43 , 14 , 1
-404 , 56 , 41 , 1
-467 , 87 , 78 , 1
-540 , 103 , 120 , 1
-626 , 102 , 170 , 1
-637 , 133 , 176 , 1
-541 , 135 , 121 , 1
-468 , 152 , 79 , 1
-403 , 184 , 41 , 1
-355 , 196 , 13 , 1
-352 , 206 , 12 , 1
-341 , 208 , 5 , 1
-335 , 218 , 2 , 1
-329 , 221 ,-2 , 1
-330 , 22 ,-1 , 2
-338 , 24 , 1 , 1
-345 , 35 , 3 , 1
-356 , 36 , 5 , 1
-359 , 43 , 6 , 1
-412 , 56 , 21 , 1
-483 , 87 , 39 , 1
-565 , 103 , 61 , 1
-660 , 102 , 87 , 1
-673 , 133 , 90 , 1
-566 , 135 , 62 , 1
-484 , 152 , 40 , 1
-411 , 184 , 20 , 1
-358 , 196 , 6 , 1
-355 , 206 , 5 , 1
-342 , 208 , 2 , 1
-336 , 218 , 0 , 1
-329 , 221 ,-2 , 1
-330 , 22 ,-1 , 2
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-330 , 22 ,-1 , 1
-338 , 24 ,-1 , 2
-338 , 24 ,-4 , 1
-337 , 24 ,-5 , 1
-336 , 24 ,-7 , 1
-334 , 24 ,-8 , 1
-332 , 24 ,-9 , 1
-330 , 24 ,-9 , 1
-328 , 24 ,-9 , 1
-326 , 24 ,-8 , 1
-324 , 24 ,-7 , 1
-323 , 24 ,-5 , 1
-322 , 24 ,-4 , 1
-322 , 24 ,-1 , 1
-322 , 24 , 1 , 1
-323 , 24 , 3 , 1
-324 , 24 , 4 , 1
-326 , 24 , 5 , 1
-328 , 24 , 6 , 1
-330 , 24 , 7 , 1
-332 , 24 , 6 , 1
-334 , 24 , 5 , 1
-336 , 24 , 4 , 1
-337 , 24 , 3 , 1
```

```
-338 , 24 , 1 , 1
-338 , 24 ,-1 , 1
-346 , 35 ,-1 , 2
-345 , 35 ,-6 , 1
-344 , 35 ,-9 , 1
-341 , 35 ,-13 , 1
-338 , 35 ,-15 , 1
-334 , 35 ,-17 , 1
-330 , 35 ,-17 , 1
-326 , 35 ,-17 , 1
-322 , 35 ,-15 , 1
-319 , 35 ,-13 , 1
-316 , 35 ,-9 , 1
-314 , 35 ,-6 , 1
-314 , 35 ,-1 , 1
-314 , 35 , 3 , 1
-316 , 35 , 7 , 1
-319 , 35 , 10 , 1
-322 , 35 , 12 , 1
-326 , 35 , 14 , 1
-330 , 35 , 15 , 1
-334 , 35 , 14 , 1
-338 , 35 , 12 , 1
-341 , 35 , 10 , 1
-344 , 35 , 7 , 1
-345 , 35 , 3 , 1
-346 , 35 ,-1 , 1
-357 , 36 ,-1 , 2
-356 , 36 ,-8 , 1
-353 , 36 ,-15 , 1
-349 , 36 ,-21 , 1
-343 , 36 ,-25 , 1
-337 , 36 ,-28 , 1
-330 , 36 ,-28 , 1
-323 , 36 ,-28 , 1
-316 , 36 ,-25 , 1
-311 , 36 ,-21 , 1
-307 , 36 ,-15 , 1
-304 , 36 ,-8 , 1
-303 , 36 ,-2 , 1
-304 , 36 , 5 , 1
-307 , 36 , 12 , 1
-311 , 36 , 18 , 1
-316 , 36 , 22 , 1
-323 , 36 , 25 , 1
-330 , 36 , 26 , 1
-337 , 36 , 25 , 1
-343 , 36 , 22 , 1
-349 , 36 , 18 , 1
-353 , 36 , 12 , 1
-356 , 36 , 5 , 1
-357 , 36 ,-1 , 1
-360 , 43 ,-1 , 2
-359 , 43 ,-9 , 1
-356 , 43 ,-16 , 1
-351 , 43 ,-23 , 1
-345 , 43 ,-27 , 1
-338 , 43 ,-30 , 1
-330 , 43 ,-31 , 1
-322 , 43 ,-30 , 1
-315 , 43 ,-27 , 1
-309 , 43 ,-23 , 1
-304 , 43 ,-16 , 1
-301 , 43 ,-9 , 1
-300 , 43 ,-2 , 1
-301 , 43 , 6 , 1
-304 , 43 , 13 , 1
-309 , 43 , 20 , 1
-315 , 43 , 24 , 1
-322 , 43 , 27 , 1
-330 , 43 , 29 , 1
-338 , 43 , 27 , 1
-345 , 43 , 24 , 1
-351 , 43 , 20 , 1
-356 , 43 , 14 , 1
-359 , 43 , 6 , 1
-360 , 43 ,-1 , 1
-415 , 56 ,-1 , 2
-412 , 56 ,-23 , 1
-404 , 56 ,-44 , 1
-390 , 56 ,-62 , 1
-372 , 56 ,-75 , 1
-352 , 56 ,-84 , 1
-330 , 56 ,-86 , 1
-308 , 56 ,-84 , 1
-287 , 56 ,-75 , 1
-270 , 56 ,-62 , 1
-256 , 56 ,-44 , 1
-248 , 56 ,-23 , 1
-245 , 56 ,-2 , 1
-248 , 56 , 20 , 1
-256 , 56 , 41 , 1
-270 , 56 , 59 , 1
-287 , 56 , 72 , 1
-308 , 56 , 81 , 1
-330 , 56 , 84 , 1
-352 , 56 , 81 , 1
-372 , 56 , 72 , 1
-390 , 56 , 59 , 1
-404 , 56 , 41 , 1
-412 , 56 , 21 , 1
-415 , 56 ,-1 , 1
-488 , 87 ,-1 , 2
-483 , 87 ,-42 , 1
-467 , 87 ,-80 , 1
-442 , 87 ,-113 , 1
-409 , 87 ,-138 , 1
-371 , 87 ,-154 , 1
-330 , 87 ,-159 , 1
-289 , 87 ,-154 , 1
-251 , 87 ,-138 , 1
-218 , 87 ,-113 , 1
-193 , 87 ,-81 , 1
-177 , 87 ,-42 , 1
-172 , 87 ,-2 , 1
-177 , 87 , 39 , 1
-193 , 87 , 77 , 1
-218 , 87 , 110 , 1
-251 , 87 , 135 , 1
-289 , 87 , 151 , 1
-330 , 87 , 157 , 1
-371 , 87 , 151 , 1
-409 , 87 , 135 , 1
-442 , 87 , 110 , 1
-467 , 87 , 78 , 1
-483 , 87 , 39 , 1
-488 , 87 ,-1 , 1
-573 , 103 ,-1 , 2
-565 , 103 ,-64 , 1
-540 , 103 ,-123 , 1
-502 , 103 ,-173 , 1
-451 , 103 ,-212 , 1
-393 , 103 ,-236 , 1
-330 , 103 ,-244 , 1
-267 , 103 ,-236 , 1
-208 , 103 ,-212 , 1
-158 , 103 ,-173 , 1
-119 , 103 ,-123 , 1
-95 , 103 ,-64 , 1
-87 , 103 ,-2 , 1
-95 , 103 , 61 , 1
-119 , 103 , 120 , 1
-158 , 103 , 170 , 1
-208 , 103 , 209 , 1
-267 , 103 , 233 , 1
-330 , 103 , 242 , 1
-393 , 103 , 233 , 1
-451 , 103 , 209 , 1
-502 , 103 , 170 , 1
-540 , 103 , 120 , 1
-565 , 103 , 61 , 1
-573 , 103 ,-1 , 1
-672 , 102 ,-1 , 2
-660 , 102 ,-90 , 1
-626 , 102 ,-172 , 1
-572 , 102 ,-243 , 1
-501 , 102 ,-298 , 1
-418 , 102 ,-332 , 1
-330 , 102 ,-343 , 1
-241 , 102 ,-332 , 1
-159 , 102 ,-298 , 1
-88 , 102 ,-243 , 1
```

```
-34 , 102 ,-173 , 1
 0 , 102 ,-90 , 1
 12 , 102 ,-2 , 1
 0 , 102 , 87 , 1
-34 , 102 , 169 , 1
-88 , 102 , 240 , 1
-159 , 102 , 295 , 1
-241 , 102 , 329 , 1
-330 , 102 , 341 , 1
-418 , 102 , 329 , 1
-501 , 102 , 295 , 1
-572 , 102 , 240 , 1
-626 , 102 , 170 , 1
-660 , 102 , 87 , 1
-672 , 102 ,-1 , 1
-685 , 133 ,-1 , 2
-673 , 133 ,-93 , 1
-637 , 133 ,-179 , 1
-581 , 133 ,-253 , 1
-507 , 133 ,-309 , 1
-422 , 133 ,-344 , 1
-330 , 133 ,-356 , 1
-238 , 133 ,-344 , 1
-152 , 133 ,-309 , 1
-79 , 133 ,-253 , 1
-22 , 133 ,-179 , 1
 13 , 133 ,-93 , 1
 25 , 133 ,-2 , 1
 13 , 133 , 90 , 1
-22 , 133 , 176 , 1
-79 , 133 , 250 , 1
-152 , 133 , 306 , 1
-238 , 133 , 341 , 1
-330 , 133 , 354 , 1
-422 , 133 , 341 , 1
-507 , 133 , 306 , 1
-581 , 133 , 250 , 1
-637 , 133 , 176 , 1
-673 , 133 , 90 , 1
-685 , 133 ,-1 , 1
-574 , 135 ,-1 , 2
-566 , 135 ,-65 , 1
-541 , 135 ,-123 , 1
-502 , 135 ,-174 , 1
-452 , 135 ,-213 , 1
-393 , 135 ,-237 , 1
-330 , 135 ,-245 , 1
-267 , 135 ,-237 , 1
-208 , 135 ,-213 , 1
-157 , 135 ,-174 , 1
-119 , 135 ,-123 , 1
-94 , 135 ,-65 , 1
-86 , 135 ,-2 , 1
-94 , 135 , 62 , 1
-119 , 135 , 121 , 1
-157 , 135 , 171 , 1
-208 , 135 , 210 , 1
-267 , 135 , 234 , 1
-330 , 135 , 243 , 1
-393 , 135 , 234 , 1
-452 , 135 , 210 , 1
-502 , 135 , 171 , 1
-541 , 135 , 121 , 1
-566 , 135 , 62 , 1
-574 , 135 ,-1 , 1
-490 , 152 ,-1 , 2
-484 , 152 ,-43 , 1
-468 , 152 ,-81 , 1
-443 , 152 ,-115 , 1
-410 , 152 ,-140 , 1
-371 , 152 ,-156 , 1
-330 , 152 ,-161 , 1
-289 , 152 ,-156 , 1
-250 , 152 ,-140 , 1
-217 , 152 ,-115 , 1
-191 , 152 ,-82 , 1
-175 , 152 ,-43 , 1
-170 , 152 ,-2 , 1
-175 , 152 , 40 , 1
-191 , 152 , 78 , 1
-217 , 152 , 112 , 1
-250 , 152 , 137 , 1
-289 , 152 , 153 , 1
-330 , 152 , 159 , 1
-371 , 152 , 153 , 1
-410 , 152 , 137 , 1
-443 , 152 , 112 , 1
-468 , 152 , 79 , 1
-484 , 152 , 40 , 1
-490 , 152 ,-1 , 1
-414 , 184 ,-1 , 2
-411 , 184 ,-23 , 1
-403 , 184 ,-43 , 1
-389 , 184 ,-61 , 1
-372 , 184 ,-74 , 1
-352 , 184 ,-83 , 1
-330 , 184 ,-85 , 1
-308 , 184 ,-83 , 1
-288 , 184 ,-74 , 1
-271 , 184 ,-61 , 1
-257 , 184 ,-44 , 1
-249 , 184 ,-23 , 1
-246 , 184 ,-2 , 1
-249 , 184 , 20 , 1
-257 , 184 , 40 , 1
-271 , 184 , 58 , 1
-288 , 184 , 71 , 1
-308 , 184 , 80 , 1
-330 , 184 , 83 , 1
-352 , 184 , 80 , 1
-372 , 184 , 71 , 1
-389 , 184 , 58 , 1
-403 , 184 , 41 , 1
-411 , 184 , 20 , 1
-414 , 184 ,-1 , 1
-359 , 196 ,-1 , 2
-358 , 196 ,-9 , 1
-355 , 196 ,-16 , 1
-350 , 196 ,-22 , 1
-344 , 196 ,-27 , 1
-337 , 196 ,-30 , 1
-330 , 196 ,-30 , 1
-322 , 196 ,-30 , 1
-315 , 196 ,-27 , 1
-309 , 196 ,-22 , 1
-305 , 196 ,-16 , 1
-302 , 196 ,-9 , 1
-301 , 196 ,-2 , 1
-302 , 196 , 6 , 1
-305 , 196 , 13 , 1
-309 , 196 , 19 , 1
-315 , 196 , 24 , 1
-322 , 196 , 27 , 1
-330 , 196 , 28 , 1
-337 , 196 , 27 , 1
-344 , 196 , 24 , 1
-350 , 196 , 19 , 1
-355 , 196 , 13 , 1
-358 , 196 , 6 , 1
-359 , 196 ,-1 , 1
-356 , 206 ,-1 , 2
-355 , 206 ,-8 , 1
-352 , 206 ,-14 , 1
-348 , 206 ,-20 , 1
-343 , 206 ,-24 , 1
-337 , 206 ,-27 , 1
-330 , 206 ,-27 , 1
-323 , 206 ,-27 , 1
-317 , 206 ,-24 , 1
-312 , 206 ,-20 , 1
-307 , 206 ,-14 , 1
-305 , 206 ,-8 , 1
-304 , 206 ,-2 , 1
-305 , 206 , 5 , 1
-307 , 206 , 11 , 1
-312 , 206 , 17 , 1
-317 , 206 , 21 , 1
-323 , 206 , 24 , 1
-330 , 206 , 25 , 1
-337 , 206 , 24 , 1
-343 , 206 , 21 , 1
-348 , 206 , 17 , 1
```

```
-352 , 206 , 12 , 1
-355 , 206 , 5 , 1
-356 , 206 ,-1 , 1
-343 , 208 ,-1 , 2
-342 , 208 ,-5 , 1
-341 , 208 ,-8 , 1
-339 , 208 ,-11 , 1
-336 , 208 ,-13 , 1
-333 , 208 ,-14 , 1
-330 , 208 ,-14 , 1
-327 , 208 ,-14 , 1
-323 , 208 ,-13 , 1
-321 , 208 ,-11 , 1
-319 , 208 ,-8 , 1
-317 , 208 ,-5 , 1
-317 , 208 ,-1 , 1
-317 , 208 , 2 , 1
-319 , 208 , 5 , 1
-321 , 208 , 8 , 1
-323 , 208 , 10 , 1
-327 , 208 , 11 , 1
-330 , 208 , 12 , 1
-333 , 208 , 11 , 1
-336 , 208 , 10 , 1
-339 , 208 , 8 , 1
-341 , 208 , 5 , 1
-342 , 208 , 2 , 1
-343 , 208 ,-1 , 1
-336 , 218 ,-1 , 2
-336 , 218 ,-3 , 1
-335 , 218 ,-4 , 1
-334 , 218 ,-6 , 1
-333 , 218 ,-7 , 1
-331 , 218 ,-7 , 1
-330 , 218 ,-7 , 1
-328 , 218 ,-7 , 1
-327 , 218 ,-7 , 1
-326 , 218 ,-6 , 1
-325 , 218 ,-4 , 1
-324 , 218 ,-3 , 1
-324 , 218 ,-1 , 1
-324 , 218 , 0 , 1
-325 , 218 , 2 , 1
-326 , 218 , 3 , 1
-327 , 218 , 4 , 1
-328 , 218 , 4 , 1
-330 , 218 , 5 , 1
-331 , 218 , 4 , 1
-333 , 218 , 4 , 1
-334 , 218 , 3 , 1
-335 , 218 , 2 , 1
-336 , 218 , 0 , 1
-336 , 218 ,-1 , 1
-329 , 221 ,-1 , 2
-329 , 221 ,-1 , 1
-329 , 221 ,-1 , 1
-329 , 221 ,-1 , 1
-329 , 221 ,-1 , 1
-330 , 221 ,-1 , 1
-330 , 221 , 0 , 1
-330 , 221 ,-1 , 1
-330 , 221 ,-1 , 1
-331 , 221 ,-1 , 1
-331 , 221 ,-1 , 1
-331 , 221 ,-1 , 1
-331 , 221 ,-1 , 1
-331 , 221 ,-2 , 1
-331 , 221 ,-2 , 1
-331 , 221 ,-2 , 1
-330 , 221 ,-2 , 1
-330 , 221 ,-2 , 1
-330 , 221 ,-2 , 1
-330 , 221 ,-2 , 1
-329 , 221 ,-2 , 1
-329 , 221 ,-2 , 1
-329 , 221 ,-2 , 1
-329 , 221 ,-2 , 1
-329 , 221 ,-1 , 1
 159 , 67 , 271 , 2
 161 , 83 , 271 , 1
 180 , 105 , 271 , 1
 197 , 110 , 271 , 1
 203 , 116 , 271 , 1
 222 , 117 , 271 , 1
 225 , 113 , 271 , 1
 521 , 106 , 271 , 1
 909 , 104 , 271 , 1
 925 , 88 , 271 , 1
 931 , 77 , 271 , 1
 932 , 67 , 271 , 1
 159 , 67 , 271 , 2
 161 , 82 , 265 , 1
 180 , 102 , 257 , 1
 197 , 107 , 255 , 1
 203 , 112 , 252 , 1
 222 , 113 , 252 , 1
 225 , 109 , 254 , 1
 521 , 103 , 256 , 1
 909 , 101 , 257 , 1
 925 , 86 , 263 , 1
 931 , 76 , 267 , 1
 932 , 67 , 271 , 1
 159 , 67 , 271 , 2
 161 , 78 , 260 , 1
 180 , 94 , 245 , 1
 197 , 98 , 241 , 1
 203 , 102 , 236 , 1
 222 , 102 , 236 , 1
 225 , 99 , 239 , 1
 521 , 95 , 244 , 1
 909 , 93 , 245 , 1
 925 , 82 , 256 , 1
 931 , 74 , 264 , 1
 932 , 67 , 271 , 1
 159 , 67 , 271 , 2
 161 , 73 , 257 , 1
 180 , 81 , 237 , 1
 197 , 84 , 231 , 1
 203 , 86 , 226 , 1
 222 , 86 , 225 , 1
 225 , 85 , 229 , 1
 521 , 82 , 235 , 1
 909 , 81 , 237 , 1
 925 , 75 , 252 , 1
 931 , 71 , 262 , 1
 932 , 67 , 271 , 1
 159 , 67 , 271 , 2
 161 , 67 , 255 , 1
 180 , 67 , 234 , 1
 197 , 67 , 228 , 1
 203 , 67 , 222 , 1
 222 , 67 , 221 , 1
 225 , 67 , 225 , 1
 521 , 67 , 232 , 1
 909 , 67 , 234 , 1
 925 , 67 , 250 , 1
 931 , 67 , 262 , 1
 932 , 67 , 272 , 1
 159 , 67 , 271 , 2
 161 , 61 , 257 , 1
 180 , 53 , 237 , 1
 197 , 51 , 231 , 1
 203 , 48 , 226 , 1
 222 , 48 , 225 , 1
 225 , 50 , 229 , 1
 521 , 52 , 235 , 1
 909 , 53 , 237 , 1
 925 , 59 , 252 , 1
 931 , 64 , 262 , 1
 932 , 67 , 271 , 1
 159 , 67 , 271 , 2
 161 , 56 , 260 , 1
 180 , 41 , 245 , 1
 197 , 37 , 241 , 1
 203 , 33 , 236 , 1
 222 , 32 , 236 , 1
 225 , 35 , 239 , 1
 521 , 40 , 244 , 1
 909 , 41 , 245 , 1
 925 , 53 , 256 , 1
 931 , 60 , 264 , 1
 932 , 67 , 271 , 1
```

159	67	271	2
161	53	265	1
180	33	257	1
197	27	255	1
203	22	252	1
222	21	252	1
225	25	254	1
521	31	256	1
909	33	257	1
925	48	263	1
931	58	267	1
932	68	271	1
159	67	271	2
161	52	271	1
180	30	271	1
197	24	271	1
203	18	271	1
222	17	271	1
225	22	271	1
521	28	271	1
909	30	271	1
925	47	271	1
931	58	271	1
932	68	271	1
159	67	271	2
161	53	277	1
180	33	285	1
197	27	288	1
203	22	290	1
222	21	290	1
225	25	289	1
521	31	286	1
909	33	285	1
925	48	279	1
931	58	275	1
932	68	271	1
159	67	271	2
161	56	282	1
180	41	298	1
197	37	302	1
203	33	306	1
222	32	306	1
225	35	303	1
521	40	299	1
909	41	297	1
925	53	286	1
931	60	278	1
932	67	271	1
159	67	271	2
161	61	286	1
180	53	306	1
197	51	311	1
203	48	316	1
222	48	317	1
225	50	313	1
521	52	307	1
909	53	305	1
925	59	290	1
931	64	280	1
932	67	271	1
159	67	271	2
161	67	287	1
180	67	308	1
197	67	314	1
203	67	320	1
222	67	321	1
225	67	317	1
521	67	310	1
909	67	308	1
925	67	292	1
931	67	281	1
932	67	271	1
159	67	271	2
161	73	286	1
180	81	306	1
197	84	311	1
203	86	316	1
222	86	317	1
225	85	313	1
521	82	307	1
909	81	305	1
925	75	290	1
931	71	280	1
932	67	271	1
159	67	271	2
161	78	282	1
180	94	298	1
197	98	302	1
203	102	306	1
222	102	306	1
225	99	303	1
521	95	299	1
909	93	297	1
925	82	286	1
931	74	278	1
932	67	271	1
159	67	271	2
161	82	277	1
180	102	285	1
197	107	288	1
203	112	290	1
222	113	290	1
225	109	289	1
521	103	286	1
909	101	285	1
925	86	279	1
931	76	275	1
932	67	271	1
159	67	271	2
159	67	271	1
159	67	271	1
159	67	271	1
159	67	271	1
159	67	271	1
159	67	271	1
159	67	271	1
159	67	271	1
159	67	271	1
159	67	271	1
159	67	271	1
159	67	271	1
159	67	271	1
159	67	271	1
159	67	271	1
159	67	271	1
161	83	271	2
161	82	265	1
161	78	260	1
161	73	257	1
161	67	255	1
161	61	257	1
161	56	260	1
161	53	265	1
161	52	271	1
161	53	277	1
161	56	282	1
161	61	286	1
161	67	287	1
161	73	286	1
161	78	282	1
161	82	277	1
161	83	271	1
180	105	271	2
180	102	257	1
180	94	245	1
180	81	237	1
180	67	234	1
180	53	237	1
180	41	245	1
180	33	257	1
180	30	271	1
180	33	285	1
180	41	298	1
180	53	306	1
180	67	308	1
180	81	306	1
180	94	298	1
180	102	285	1
180	105	271	1
197	110	271	2
197	107	255	1
197	98	241	1

```
197 , 84 , 231 , 1
197 , 67 , 228 , 1
197 , 51 , 231 , 1
197 , 37 , 241 , 1
197 , 27 , 255 , 1
197 , 24 , 271 , 1
197 , 27 , 288 , 1
197 , 37 , 302 , 1
197 , 51 , 311 , 1
197 , 67 , 314 , 1
197 , 84 , 311 , 1
197 , 98 , 302 , 1
197 , 107 , 288 , 1
197 , 110 , 271 , 1
203 , 116 , 271 , 2
203 , 112 , 252 , 1
203 , 102 , 236 , 1
203 , 86 , 226 , 1
203 , 67 , 222 , 1
203 , 48 , 226 , 1
203 , 33 , 236 , 1
203 , 22 , 252 , 1
203 , 18 , 271 , 1
203 , 22 , 290 , 1
203 , 33 , 306 , 1
203 , 48 , 316 , 1
203 , 67 , 320 , 1
203 , 86 , 316 , 1
203 , 102 , 306 , 1
203 , 112 , 290 , 1
203 , 116 , 271 , 1
222 , 117 , 271 , 2
222 , 113 , 252 , 1
222 , 102 , 236 , 1
222 , 86 , 225 , 1
222 , 67 , 221 , 1
222 , 48 , 225 , 1
222 , 32 , 236 , 1
222 , 21 , 252 , 1
222 , 17 , 271 , 1
222 , 21 , 290 , 1
222 , 32 , 306 , 1
222 , 48 , 317 , 1
222 , 67 , 321 , 1
222 , 86 , 317 , 1
222 , 102 , 306 , 1
222 , 113 , 290 , 1
222 , 117 , 271 , 1
225 , 113 , 271 , 2
225 , 109 , 254 , 1
225 , 99 , 239 , 1
225 , 85 , 229 , 1
225 , 67 , 225 , 1
225 , 50 , 229 , 1
225 , 35 , 239 , 1
225 , 25 , 254 , 1
225 , 22 , 271 , 1
225 , 25 , 289 , 1
225 , 35 , 303 , 1
225 , 50 , 313 , 1
225 , 67 , 317 , 1
225 , 85 , 313 , 1
225 , 99 , 303 , 1
225 , 109 , 289 , 1
225 , 113 , 271 , 1
521 , 106 , 271 , 2
521 , 103 , 256 , 1
521 , 95 , 244 , 1
521 , 82 , 235 , 1
521 , 67 , 232 , 1
521 , 52 , 235 , 1
521 , 40 , 244 , 1
521 , 31 , 256 , 1
521 , 28 , 271 , 1
521 , 31 , 286 , 1
521 , 40 , 299 , 1
521 , 52 , 307 , 1
521 , 67 , 310 , 1
521 , 82 , 307 , 1
521 , 95 , 299 , 1
521 , 103 , 286 , 1
521 , 106 , 271 , 1
909 , 104 , 271 , 2
909 , 101 , 257 , 1
909 , 93 , 245 , 1
909 , 81 , 237 , 1
909 , 67 , 234 , 1
909 , 53 , 237 , 1
909 , 41 , 245 , 1
909 , 33 , 257 , 1
909 , 30 , 271 , 1
909 , 33 , 285 , 1
909 , 41 , 297 , 1
909 , 53 , 305 , 1
909 , 67 , 308 , 1
909 , 81 , 305 , 1
909 , 93 , 297 , 1
909 , 101 , 285 , 1
909 , 104 , 271 , 1
925 , 88 , 271 , 2
925 , 86 , 263 , 1
925 , 82 , 256 , 1
925 , 75 , 252 , 1
925 , 67 , 250 , 1
925 , 59 , 252 , 1
925 , 53 , 256 , 1
925 , 48 , 263 , 1
925 , 47 , 271 , 1
925 , 48 , 279 , 1
925 , 53 , 286 , 1
925 , 59 , 290 , 1
925 , 67 , 292 , 1
925 , 75 , 290 , 1
925 , 82 , 286 , 1
925 , 86 , 279 , 1
925 , 88 , 271 , 1
931 , 77 , 271 , 2
931 , 76 , 267 , 1
931 , 74 , 264 , 1
931 , 71 , 262 , 1
931 , 67 , 262 , 1
931 , 64 , 262 , 1
931 , 60 , 264 , 1
931 , 58 , 267 , 1
931 , 58 , 271 , 1
931 , 58 , 275 , 1
931 , 60 , 278 , 1
931 , 64 , 280 , 1
931 , 67 , 281 , 1
931 , 71 , 280 , 1
931 , 74 , 278 , 1
931 , 76 , 275 , 1
931 , 77 , 271 , 1
932 , 67 , 271 , 2
932 , 67 , 271 , 1
932 , 67 , 271 , 1
932 , 67 , 271 , 1
932 , 67 , 272 , 1
932 , 67 , 271 , 1
932 , 67 , 271 , 1
932 , 68 , 271 , 1
932 , 68 , 271 , 1
932 , 68 , 271 , 1
932 , 67 , 271 , 1
932 , 67 , 271 , 1
932 , 67 , 271 , 1
932 , 67 , 271 , 1
932 , 67 , 271 , 1
932 , 67 , 271 , 1
932 , 67 , 271 , 1
158 , 66 ,-265 , 2
160 , 82 ,-265 , 1
179 , 103 ,-265 , 1
196 , 109 ,-265 , 1
202 , 115 ,-265 , 1
221 , 116 ,-265 , 1
224 , 112 ,-265 , 1
520 , 105 ,-265 , 1
908 , 103 ,-265 , 1
924 , 87 ,-265 , 1
930 , 76 ,-265 , 1
931 , 66 ,-265 , 1
```

```
158 , 66 ,-265 , 2
160 , 81 ,-271 , 1
179 , 101 ,-280 , 1
196 , 106 ,-282 , 1
202 , 111 ,-284 , 1
221 , 112 ,-285 , 1
224 , 108 ,-283 , 1
520 , 102 ,-280 , 1
908 , 100 ,-280 , 1
924 , 85 ,-273 , 1
930 , 75 ,-269 , 1
931 , 66 ,-265 , 1
158 , 66 ,-265 , 2
160 , 77 ,-277 , 1
179 , 93 ,-292 , 1
196 , 97 ,-296 , 1
202 , 101 ,-300 , 1
221 , 101 ,-301 , 1
224 , 98 ,-298 , 1
520 , 94 ,-293 , 1
908 , 92 ,-292 , 1
924 , 81 ,-280 , 1
930 , 73 ,-272 , 1
931 , 66 ,-265 , 1
158 , 66 ,-265 , 2
160 , 72 ,-280 , 1
179 , 80 ,-300 , 1
196 , 83 ,-305 , 1
202 , 85 ,-311 , 1
221 , 85 ,-311 , 1
224 , 84 ,-308 , 1
520 , 81 ,-302 , 1
908 , 80 ,-300 , 1
924 , 74 ,-285 , 1
930 , 70 ,-274 , 1
931 , 66 ,-265 , 1
158 , 66 ,-265 , 2
160 , 66 ,-281 , 1
179 , 66 ,-303 , 1
196 , 66 ,-309 , 1
202 , 66 ,-315 , 1
221 , 66 ,-315 , 1
224 , 66 ,-311 , 1
520 , 66 ,-305 , 1
908 , 66 ,-302 , 1
924 , 66 ,-286 , 1
930 , 66 ,-275 , 1
931 , 66 ,-265 , 1
158 , 66 ,-265 , 2
160 , 60 ,-280 , 1
179 , 52 ,-300 , 1
196 , 50 ,-305 , 1
202 , 47 ,-311 , 1
221 , 47 ,-311 , 1
```

```
224 , 49 ,-308 , 1
520 , 51 ,-302 , 1
908 , 52 ,-300 , 1
924 , 58 ,-285 , 1
930 , 62 ,-274 , 1
931 , 66 ,-265 , 1
158 , 66 ,-265 , 2
160 , 55 ,-277 , 1
179 , 40 ,-292 , 1
196 , 36 ,-296 , 1
202 , 31 ,-300 , 1
221 , 31 ,-301 , 1
224 , 34 ,-298 , 1
520 , 39 ,-293 , 1
908 , 40 ,-292 , 1
924 , 51 ,-280 , 1
930 , 59 ,-272 , 1
931 , 66 ,-265 , 1
158 , 66 ,-265 , 2
160 , 52 ,-271 , 1
179 , 32 ,-280 , 1
196 , 26 ,-282 , 1
202 , 21 ,-284 , 1
221 , 20 ,-285 , 1
224 , 24 ,-283 , 1
520 , 30 ,-280 , 1
908 , 32 ,-280 , 1
924 , 47 ,-273 , 1
930 , 57 ,-269 , 1
931 , 66 ,-265 , 1
158 , 66 ,-265 , 2
160 , 50 ,-265 , 1
179 , 29 ,-265 , 1
196 , 23 ,-265 , 1
202 , 17 ,-265 , 1
221 , 16 ,-265 , 1
224 , 20 ,-265 , 1
520 , 27 ,-265 , 1
908 , 29 ,-265 , 1
924 , 45 ,-265 , 1
930 , 57 ,-265 , 1
931 , 67 ,-265 , 1
158 , 66 ,-266 , 2
160 , 52 ,-260 , 1
179 , 32 ,-251 , 1
196 , 26 ,-249 , 1
202 , 21 ,-247 , 1
221 , 20 ,-246 , 1
224 , 24 ,-248 , 1
520 , 30 ,-251 , 1
908 , 32 ,-251 , 1
924 , 47 ,-258 , 1
930 , 57 ,-262 , 1
931 , 66 ,-266 , 1
```

```
158 , 66 ,-266 , 2
160 , 55 ,-254 , 1
179 , 40 ,-239 , 1
196 , 36 ,-235 , 1
202 , 31 ,-231 , 1
221 , 31 ,-230 , 1
224 , 34 ,-233 , 1
520 , 39 ,-238 , 1
908 , 40 ,-239 , 1
924 , 51 ,-251 , 1
930 , 59 ,-259 , 1
931 , 66 ,-266 , 1
158 , 66 ,-266 , 2
160 , 60 ,-251 , 1
179 , 52 ,-231 , 1
196 , 50 ,-226 , 1
202 , 47 ,-220 , 1
221 , 47 ,-220 , 1
224 , 49 ,-223 , 1
520 , 51 ,-229 , 1
908 , 52 ,-231 , 1
924 , 58 ,-246 , 1
930 , 62 ,-257 , 1
931 , 66 ,-266 , 1
158 , 66 ,-266 , 2
160 , 66 ,-250 , 1
179 , 66 ,-228 , 1
196 , 66 ,-222 , 1
202 , 66 ,-216 , 1
221 , 66 ,-216 , 1
224 , 66 ,-220 , 1
520 , 66 ,-226 , 1
908 , 66 ,-229 , 1
924 , 66 ,-245 , 1
930 , 66 ,-256 , 1
931 , 66 ,-266 , 1
158 , 66 ,-266 , 2
160 , 72 ,-251 , 1
179 , 80 ,-231 , 1
196 , 83 ,-226 , 1
202 , 85 ,-220 , 1
221 , 85 ,-220 , 1
224 , 84 ,-223 , 1
520 , 81 ,-229 , 1
908 , 80 ,-231 , 1
924 , 74 ,-246 , 1
930 , 70 ,-257 , 1
931 , 66 ,-266 , 1
158 , 66 ,-266 , 2
160 , 77 ,-254 , 1
179 , 93 ,-239 , 1
196 , 97 ,-235 , 1
202 , 101 ,-231 , 1
221 , 101 ,-230 , 1
```

```
224 , 98 ,-233 , 1
520 , 94 ,-238 , 1
908 , 92 ,-239 , 1
924 , 81 ,-251 , 1
930 , 73 ,-259 , 1
931 , 66 ,-266 , 1
158 , 66 ,-266 , 2
160 , 81 ,-260 , 1
179 , 101 ,-251 , 1
196 , 106 ,-249 , 1
202 , 111 ,-247 , 1
221 , 112 ,-246 , 1
224 , 108 ,-248 , 1
520 , 102 ,-251 , 1
908 , 100 ,-251 , 1
924 , 85 ,-258 , 1
930 , 75 ,-262 , 1
931 , 66 ,-266 , 1
158 , 66 ,-265 , 2
158 , 66 ,-265 , 1
158 , 66 ,-265 , 1
158 , 66 ,-265 , 1
158 , 66 ,-265 , 1
158 , 66 ,-265 , 1
158 , 66 ,-265 , 1
158 , 66 ,-265 , 1
158 , 66 ,-265 , 1
158 , 66 ,-266 , 1
158 , 66 ,-266 , 1
158 , 66 ,-266 , 1
158 , 66 ,-266 , 1
158 , 66 ,-266 , 1
158 , 66 ,-266 , 1
158 , 66 ,-266 , 1
158 , 66 ,-265 , 1
160 , 82 ,-265 , 2
160 , 81 ,-271 , 1
160 , 77 ,-277 , 1
160 , 72 ,-280 , 1
160 , 66 ,-281 , 1
160 , 60 ,-280 , 1
160 , 55 ,-277 , 1
160 , 52 ,-271 , 1
160 , 50 ,-265 , 1
160 , 52 ,-260 , 1
160 , 55 ,-254 , 1
160 , 60 ,-251 , 1
160 , 66 ,-250 , 1
160 , 72 ,-251 , 1
160 , 77 ,-254 , 1
160 , 81 ,-260 , 1
160 , 82 ,-265 , 1
179 , 103 ,-265 , 2
179 , 101 ,-280 , 1
179 , 93 ,-292 , 1
179 , 80 ,-300 , 1
179 , 66 ,-303 , 1
179 , 52 ,-300 , 1
179 , 40 ,-292 , 1
179 , 32 ,-280 , 1
179 , 29 ,-265 , 1
179 , 32 ,-251 , 1
179 , 40 ,-239 , 1
179 , 52 ,-231 , 1
179 , 66 ,-228 , 1
179 , 80 ,-231 , 1
179 , 93 ,-239 , 1
179 , 101 ,-251 , 1
179 , 103 ,-265 , 1
196 , 109 ,-265 , 2
196 , 106 ,-282 , 1
196 , 97 ,-296 , 1
196 , 83 ,-305 , 1
196 , 66 ,-309 , 1
196 , 50 ,-305 , 1
196 , 36 ,-296 , 1
196 , 26 ,-282 , 1
196 , 23 ,-265 , 1
196 , 26 ,-249 , 1
196 , 36 ,-235 , 1
196 , 50 ,-226 , 1
196 , 66 ,-222 , 1
196 , 83 ,-226 , 1
196 , 97 ,-235 , 1
196 , 106 ,-249 , 1
196 , 109 ,-265 , 1
202 , 115 ,-265 , 2
202 , 111 ,-284 , 1
202 , 101 ,-300 , 1
202 , 85 ,-311 , 1
202 , 66 ,-315 , 1
202 , 47 ,-311 , 1
202 , 31 ,-300 , 1
202 , 21 ,-284 , 1
202 , 17 ,-265 , 1
202 , 21 ,-247 , 1
202 , 31 ,-231 , 1
202 , 47 ,-220 , 1
202 , 66 ,-216 , 1
202 , 85 ,-220 , 1
202 , 101 ,-231 , 1
202 , 111 ,-247 , 1
202 , 115 ,-265 , 1
221 , 116 ,-265 , 2
221 , 112 ,-285 , 1
221 , 101 ,-301 , 1
221 , 85 ,-311 , 1
221 , 66 ,-315 , 1
221 , 47 ,-311 , 1
221 , 31 ,-301 , 1
221 , 20 ,-285 , 1
221 , 16 ,-265 , 1
221 , 20 ,-246 , 1
221 , 31 ,-230 , 1
221 , 47 ,-220 , 1
221 , 66 ,-216 , 1
221 , 85 ,-220 , 1
221 , 101 ,-230 , 1
221 , 112 ,-246 , 1
221 , 116 ,-265 , 1
224 , 112 ,-265 , 2
224 , 108 ,-283 , 1
224 , 98 ,-298 , 1
224 , 84 ,-308 , 1
224 , 66 ,-311 , 1
224 , 49 ,-308 , 1
224 , 34 ,-298 , 1
224 , 24 ,-283 , 1
224 , 20 ,-265 , 1
224 , 24 ,-248 , 1
224 , 34 ,-233 , 1
224 , 49 ,-223 , 1
224 , 66 ,-220 , 1
224 , 84 ,-223 , 1
224 , 98 ,-233 , 1
224 , 108 ,-248 , 1
224 , 112 ,-265 , 1
520 , 105 ,-265 , 2
520 , 102 ,-280 , 1
520 , 94 ,-293 , 1
520 , 81 ,-302 , 1
520 , 66 ,-305 , 1
520 , 51 ,-302 , 1
520 , 39 ,-293 , 1
520 , 30 ,-280 , 1
520 , 27 ,-265 , 1
520 , 30 ,-251 , 1
520 , 39 ,-238 , 1
520 , 51 ,-229 , 1
520 , 66 ,-226 , 1
520 , 81 ,-229 , 1
520 , 94 ,-238 , 1
520 , 102 ,-251 , 1
520 , 105 ,-265 , 1
908 , 103 ,-265 , 2
908 , 100 ,-280 , 1
908 , 92 ,-292 , 1
908 , 80 ,-300 , 1
908 , 66 ,-302 , 1
908 , 52 ,-300 , 1
908 , 40 ,-292 , 1
908 , 32 ,-280 , 1
```

```
908 , 29 ,-265 , 1
908 , 32 ,-251 , 1
908 , 40 ,-239 , 1
908 , 52 ,-231 , 1
908 , 66 ,-229 , 1
908 , 80 ,-231 , 1
908 , 92 ,-239 , 1
908 , 100 ,-251 , 1
908 , 103 ,-265 , 1
924 , 87 ,-265 , 2
924 , 85 ,-273 , 1
924 , 81 ,-280 , 1
924 , 74 ,-285 , 1
924 , 66 ,-286 , 1
924 , 58 ,-285 , 1
924 , 51 ,-280 , 1
924 , 47 ,-273 , 1
924 , 45 ,-265 , 1
924 , 47 ,-258 , 1
924 , 51 ,-251 , 1
924 , 58 ,-246 , 1
924 , 66 ,-245 , 1
924 , 74 ,-246 , 1
924 , 81 ,-251 , 1
924 , 85 ,-258 , 1
924 , 87 ,-265 , 1
930 , 76 ,-265 , 2
930 , 75 ,-269 , 1
930 , 73 ,-272 , 1
930 , 70 ,-274 , 1
930 , 66 ,-275 , 1
930 , 62 ,-274 , 1
930 , 59 ,-272 , 1
930 , 57 ,-269 , 1
930 , 57 ,-265 , 1
930 , 57 ,-262 , 1
930 , 59 ,-259 , 1
930 , 62 ,-257 , 1
930 , 66 ,-256 , 1
930 , 70 ,-257 , 1
930 , 73 ,-259 , 1
930 , 75 ,-262 , 1
930 , 76 ,-265 , 1
931 , 66 ,-265 , 2
931 , 66 ,-265 , 1
931 , 66 ,-265 , 1
931 , 66 ,-265 , 1
931 , 66 ,-265 , 1
931 , 66 ,-265 , 1
931 , 66 ,-265 , 1
931 , 66 ,-265 , 1
931 , 67 ,-265 , 1
931 , 66 ,-266 , 1
931 , 66 ,-266 , 1
931 , 66 ,-266 , 1
931 , 66 ,-266 , 1
931 , 66 ,-266 , 1
931 , 66 ,-266 , 1
931 , 66 ,-266 , 1
931 , 66 ,-265 , 1
-54 ,-62 ,-1 , 2
-51 , 9 ,-1 , 1
164 , 4 ,-1 , 1
288 , 0 ,-1 , 1
410 ,-14 ,-1 , 1
520 ,-21 ,-1 , 1
539 ,-24 ,-1 , 1
550 ,-22 ,-1 , 1
548 ,-66 ,-1 , 1
-54 ,-62 ,-2 , 2
-51 , 3 ,-29 , 1
164 ,-1 ,-27 , 1
288 ,-5 ,-26 , 1
410 ,-18 ,-20 , 1
520 ,-25 ,-17 , 1
539 ,-27 ,-16 , 1
550 ,-25 ,-17 , 1
548 ,-66 , 0 , 1
-54 ,-63 ,-3 , 2
-51 ,-13 ,-53 , 1
164 ,-16 ,-49 , 1
288 ,-19 ,-47 , 1
410 ,-29 ,-37 , 1
520 ,-34 ,-31 , 1
539 ,-36 ,-30 , 1
550 ,-34 ,-31 , 1
548 ,-65 , 0 , 1
-54 ,-64 ,-3 , 2
-51 ,-37 ,-69 , 1
164 ,-39 ,-64 , 1
288 ,-40 ,-61 , 1
410 ,-45 ,-48 , 1
520 ,-48 ,-41 , 1
539 ,-49 ,-39 , 1
550 ,-48 ,-41 , 1
548 ,-65 , 0 , 1
-54 ,-65 ,-3 , 2
-51 ,-65 ,-74 , 1
164 ,-65 ,-70 , 1
288 ,-65 ,-66 , 1
410 ,-65 ,-52 , 1
520 ,-65 ,-44 , 1
539 ,-65 ,-42 , 1
550 ,-65 ,-44 , 1
548 ,-65 , 0 , 1
-54 ,-66 ,-3 , 2
-51 ,-93 ,-69 , 1
164 ,-92 ,-64 , 1
288 ,-90 ,-61 , 1
410 ,-85 ,-48 , 1
520 ,-82 ,-41 , 1
539 ,-81 ,-39 , 1
550 ,-82 ,-41 , 1
548 ,-65 , 0 , 1
-54 ,-67 ,-3 , 2
-51 ,-117 ,-53 , 1
164 ,-114 ,-49 , 1
288 ,-111 ,-47 , 1
410 ,-101 ,-37 , 1
520 ,-96 ,-31 , 1
539 ,-94 ,-30 , 1
550 ,-96 ,-31 , 1
548 ,-65 , 0 , 1
-54 ,-68 ,-2 , 2
-51 ,-133 ,-29 , 1
164 ,-129 ,-27 , 1
288 ,-125 ,-26 , 1
410 ,-113 ,-20 , 1
520 ,-105 ,-17 , 1
539 ,-103 ,-16 , 1
550 ,-105 ,-17 , 1
548 ,-65 , 0 , 1
-54 ,-68 ,-1 , 2
-51 ,-139 ,-1 , 1
164 ,-134 ,-1 , 1
288 ,-130 ,-1 , 1
410 ,-116 ,-1 , 1
520 ,-109 ,-1 , 1
539 ,-107 ,-1 , 1
550 ,-108 ,-1 , 1
548 ,-64 ,-1 , 1
-54 ,-68 , 1 , 2
-51 ,-133 , 28 , 1
164 ,-129 , 26 , 1
288 ,-125 , 25 , 1
410 ,-113 , 19 , 1
520 ,-105 , 16 , 1
539 ,-103 , 15 , 1
550 ,-105 , 16 , 1
548 ,-65 ,-1 , 1
-54 ,-67 , 2 , 2
-51 ,-117 , 52 , 1
164 ,-114 , 48 , 1
288 ,-111 , 46 , 1
410 ,-101 , 36 , 1
520 ,-96 , 30 , 1
539 ,-94 , 29 , 1
550 ,-96 , 30 , 1
548 ,-65 ,-1 , 1
-54 ,-66 , 2 , 2
-51 ,-93 , 68 , 1
164 ,-92 , 63 , 1
```

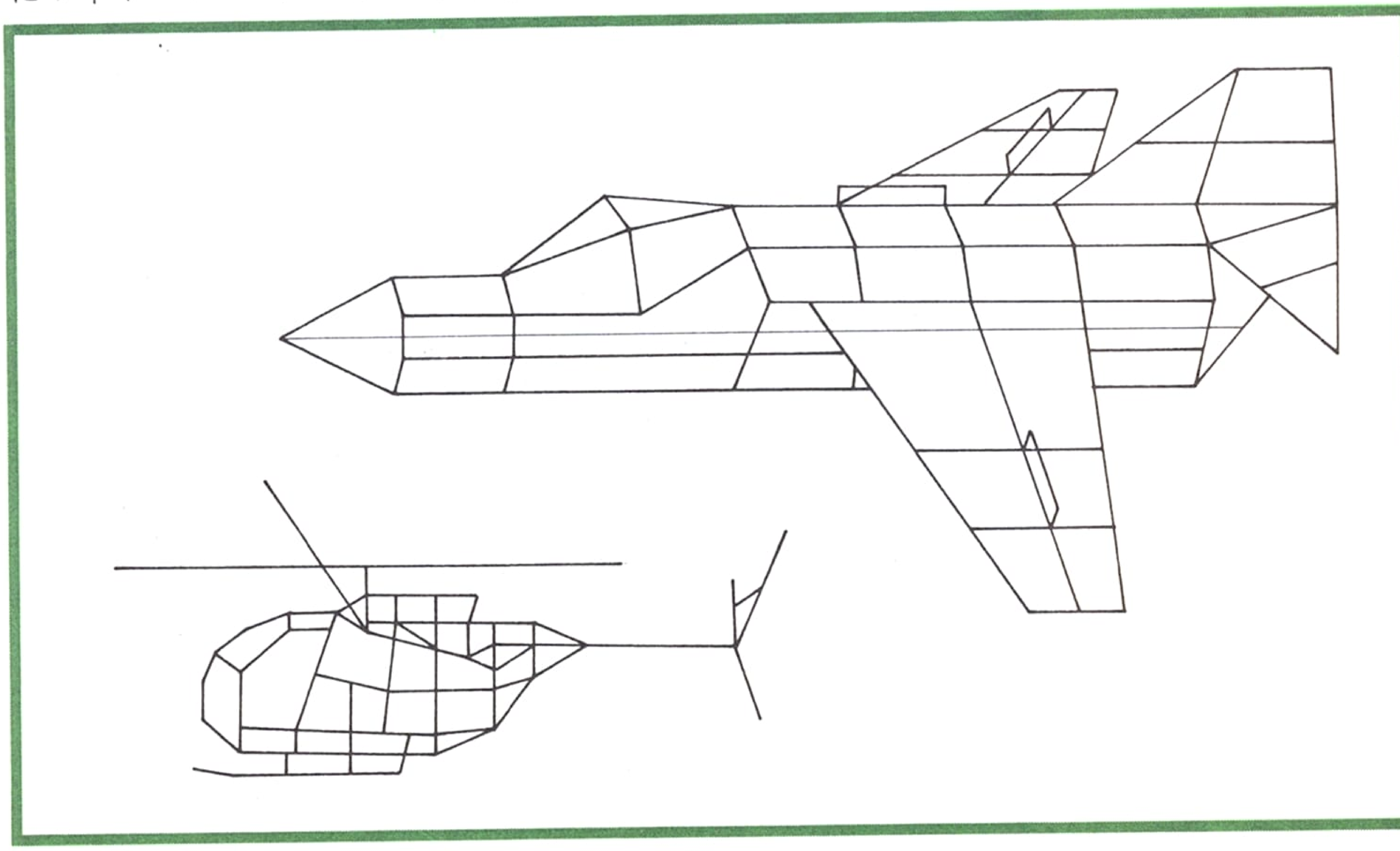
```
288 ,-90 , 60 , 1
410 ,-85 , 47 , 1
520 ,-82 , 40 , 1
539 ,-81 , 38 , 1
550 ,-82 , 40 , 1
548 ,-65 ,-1 , 1
-54 ,-65 , 2 , 2
-51 ,-65 , 73 , 1
164 ,-65 , 69 , 1
288 ,-65 , 65 , 1
410 ,-65 , 51 , 1
520 ,-65 , 43 , 1
539 ,-65 , 41 , 1
550 ,-65 , 43 , 1
548 ,-65 ,-1 , 1
-54 ,-64 , 2 , 2
-51 ,-37 , 68 , 1
164 ,-39 , 63 , 1
288 ,-40 , 60 , 1
410 ,-45 , 47 , 1
520 ,-48 , 40 , 1
539 ,-49 , 38 , 1
550 ,-48 , 40 , 1
548 ,-65 ,-1 , 1
-54 ,-63 , 2 , 2
-51 ,-13 , 52 , 1
164 ,-16 , 48 , 1
288 ,-19 , 46 , 1
410 ,-29 , 36 , 1
520 ,-34 , 30 , 1
539 ,-36 , 29 , 1
550 ,-34 , 30 , 1
548 ,-65 ,-1 , 1
-54 ,-62 , 1 , 2
-51 , 3 , 28 , 1
164 ,-1 , 26 , 1
288 ,-5 , 25 , 1
410 ,-18 , 19 , 1
520 ,-25 , 16 , 1
539 ,-27 , 15 , 1
550 ,-25 , 16 , 1
548 ,-66 ,-1 , 1
-54 ,-62 ,-1 , 2
-54 ,-62 ,-2 , 1
-54 ,-63 ,-3 , 1
-54 ,-64 ,-3 , 1
-54 ,-65 ,-3 , 1
-54 ,-66 ,-3 , 1
-54 ,-67 ,-3 , 1
-54 ,-68 ,-2 , 1
-54 ,-68 ,-1 , 1
-54 ,-68 , 1 , 1
-54 ,-67 , 2 , 1
-54 ,-66 , 2 , 1
-54 ,-65 , 2 , 1
-54 ,-64 , 2 , 1
-54 ,-63 , 2 , 1
-54 ,-62 , 1 , 1
-54 ,-62 ,-1 , 1
-51 , 9 ,-1 , 2
-51 , 3 ,-29 , 1
-51 ,-13 ,-53 , 1
-51 ,-37 ,-69 , 1
-51 ,-65 ,-74 , 1
-51 ,-93 ,-69 , 1
-51 ,-117 ,-53 , 1
-51 ,-133 ,-29 , 1
-51 ,-139 ,-1 , 1
-51 ,¬133 , 28 , 1
-51 ,-117 , 52 , 1
-51 ,-93 , 68 , 1
-51 ,-65 , 73 , 1
-51 ,-37 , 68 , 1
-51 ,-13 , 52 , 1
-51 , 3 , 28 , 1
-51 , 9 ,-1 , 1
164 , 4 ,-1 , 2
164 ,-1 ,-27 , 1
164 ,-16 ,-49 , 1
164 ,-39 ,-64 , 1
164 ,-65 ,-70 , 1
164 ,-92 ,-64 , 1
164 ,-114 ,-49 , 1
164 ,-129 ,-27 , 1
164 ,-134 ,-1 , 1
164 ,-129 , 26 , 1
164 ,-114 , 48 , 1
164 ,-92 , 63 , 1
164 ,-65 , 69 , 1
164 ,-39 , 63 , 1
164 ,-16 , 48 , 1
164 ,-1 , 26 , 1
164 , 4 ,-1 , 1
288 , 0 ,-1 , 2
288 ,-5 ,-26 , 1
288 ,-19 ,-47 , 1
288 ,-40 ,-61 , 1
288 ,-65 ,-66 , 1
288 ,-90 ,-61 , 1
288 ,-111 ,-47 , 1
288 ,-125 ,-26 , 1
288 ,-130 ,-1 , 1
288 ,-125 , 25 , 1
288 ,-111 , 46 , 1
288 ,-90 , 60 , 1
288 ,-65 , 65 , 1
288 ,-40 , 60 , 1
288 ,-19 , 46 , 1
288 ,-5 , 25 , 1
288 , 0 ,-1 , 1
410 ,-14 ,-1 , 2
410 ,-18 ,-20 , 1
410 ,-29 ,-37 , 1
410 ,-45 ,-48 , 1
410 ,-65 ,-52 , 1
410 ,-85 ,-48 , 1
410 ,-101 ,-37 , 1
410 ,-113 ,-20 , 1
410 ,-116 ,-1 , 1
410 ,-113 , 19 , 1
410 ,-101 , 36 , 1
410 ,-85 , 47 , 1
410 ,-65 , 51 , 1
410 ,-45 , 47 , 1
410 ,-29 , 36 , 1
410 ,-18 , 19 , 1
410 ,-14 ,-1 , 1
520 ,-21 ,-1 , 2
520 ,-25 ,-17 , 1
520 ,-34 ,-31 , 1
520 ,-48 ,-41 , 1
520 ,-65 ,-44 , 1
520 ,-82 ,-41 , 1
520 ,-96 ,-31 , 1
520 ,-105 ,-17 , 1
520 ,-109 ,-1 , 1
520 ,-105 , 16 , 1
520 ,-96 , 30 , 1
520 ,-82 , 40 , 1
520 ,-65 , 43 , 1
520 ,-48 , 40 , 1
520 ,-34 , 30 , 1
520 ,-25 , 16 , 1
520 ,-21 ,-1 , 1
539 ,-24 ,-1 , 2
539 ,-27 ,-16 , 1
539 ,-36 ,-30 , 1
539 ,-49 ,-39 , 1
539 ,-65 ,-42 , 1
539 ,-81 ,-39 , 1
539 ,-94 ,-30 , 1
539 ,-103 ,-16 , 1
539 ,-107 ,-1 , 1
539 ,-103 , 15 , 1
539 ,-94 , 29 , 1
539 ,-81 , 38 , 1
539 ,-65 , 41 , 1
539 ,-49 , 38 , 1
539 ,-36 , 29 , 1
539 ,-27 , 15 , 1
539 ,-24 ,-1 , 1
550 ,-22 ,-1 , 2
```

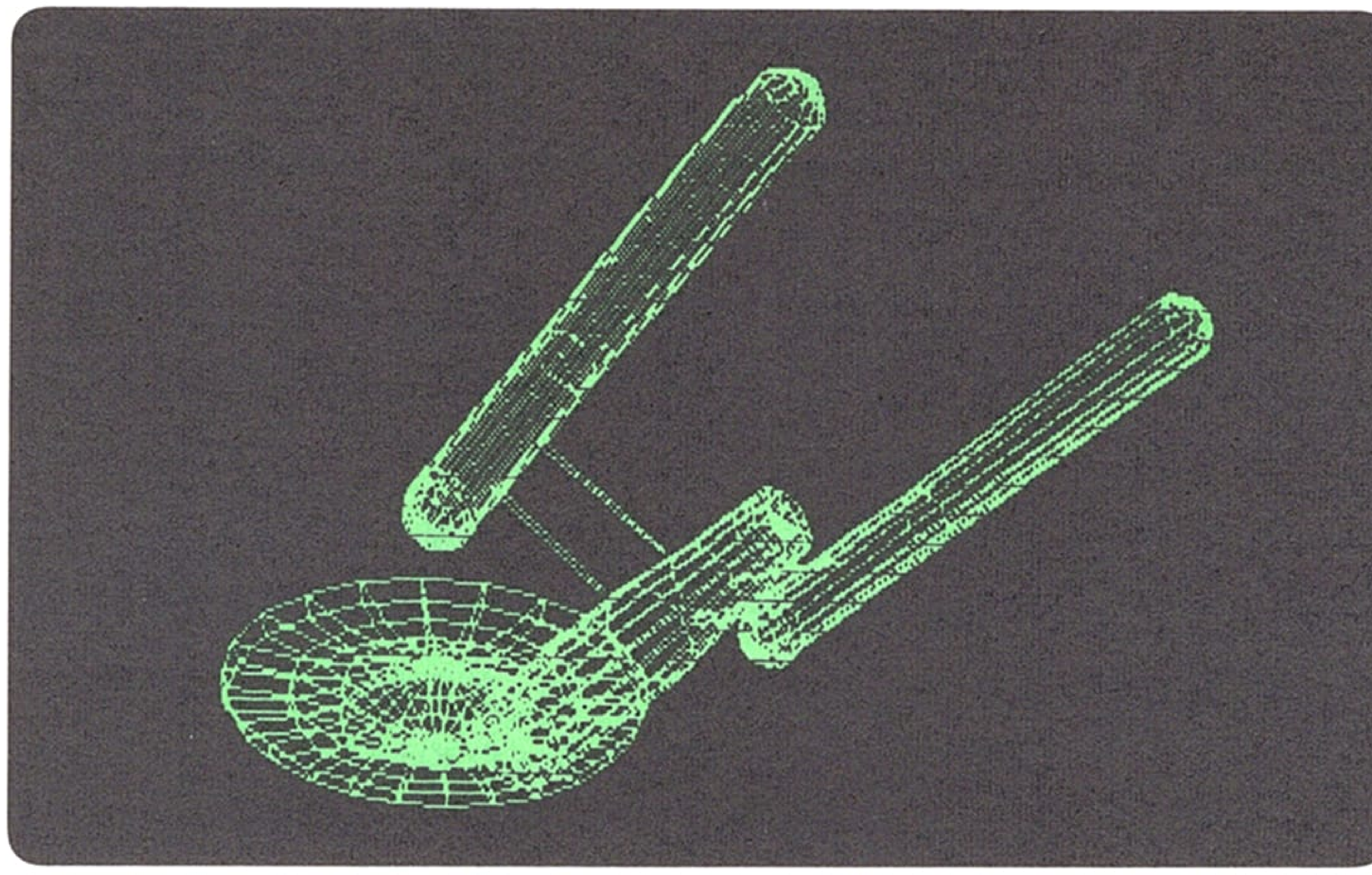
550 ,-25 ,-17 , 1	548 ,-65 , 0 , 1	291 , 62 , 228 , 1
550 ,-34 ,-31 , 1	548 ,-65 , 0 , 1	400 ,-37 , 47 , 2
550 ,-48 ,-41 , 1	548 ,-65 , 0 , 1	397 , 62 , 226 , 1
550 ,-65 ,-44 , 1	548 ,-65 , 0 , 1	290 ,-34 ,-74 , 2
550 ,-82 ,-41 , 1	548 ,-65 , 0 , 1	292 , 54 ,-233 , 1
550 ,-96 ,-31 , 1	548 ,-64 ,-1 , 1	400 ,-36 ,-58 , 2
550 ,-105 ,-17 , 1	548 ,-65 ,-1 , 1	398 , 56 ,-232 , 1
550 ,-108 ,-1 , 1	548 ,-65 ,-1 , 1	289 ,-25 ,-69 , 2
550 ,-105 , 16 , 1	548 ,-65 ,-1 , 1	291 , 65 ,-227 , 1
550 ,-96 , 30 , 1	548 ,-65 ,-1 , 1	400 ,-27 ,-56 , 2
550 ,-82 , 40 , 1	548 ,-65 ,-1 , 1	399 , 64 ,-226 , 1
550 ,-65 , 43 , 1	548 ,-65 ,-1 , 1	-4 , 0 , 17 , 2
550 ,-48 , 40 , 1	548 ,-66 ,-1 , 1	-148 , 94 , 18 , 1
550 ,-34 , 30 , 1	548 ,-66 ,-1 , 1	137 ,-1 , 18 , 2
550 ,-25 , 16 , 1	293 ,-35 , 68 , 2	9 , 102 , 20 , 1
550 ,-22 ,-1 , 1	292 , 52 , 232 , 1	-1 ,-1 ,-24 , 2
548 ,-66 ,-1 , 2	401 ,-35 , 54 , 2	-147 , 92 ,-24 , 1
548 ,-66 , 0 , 1	399 , 57 , 233 , 1	139 ,-2 ,-25 , 2
548 ,-65 , 0 , 1	292 ,-25 , 62 , 2	11 , 101 ,-24 , 1

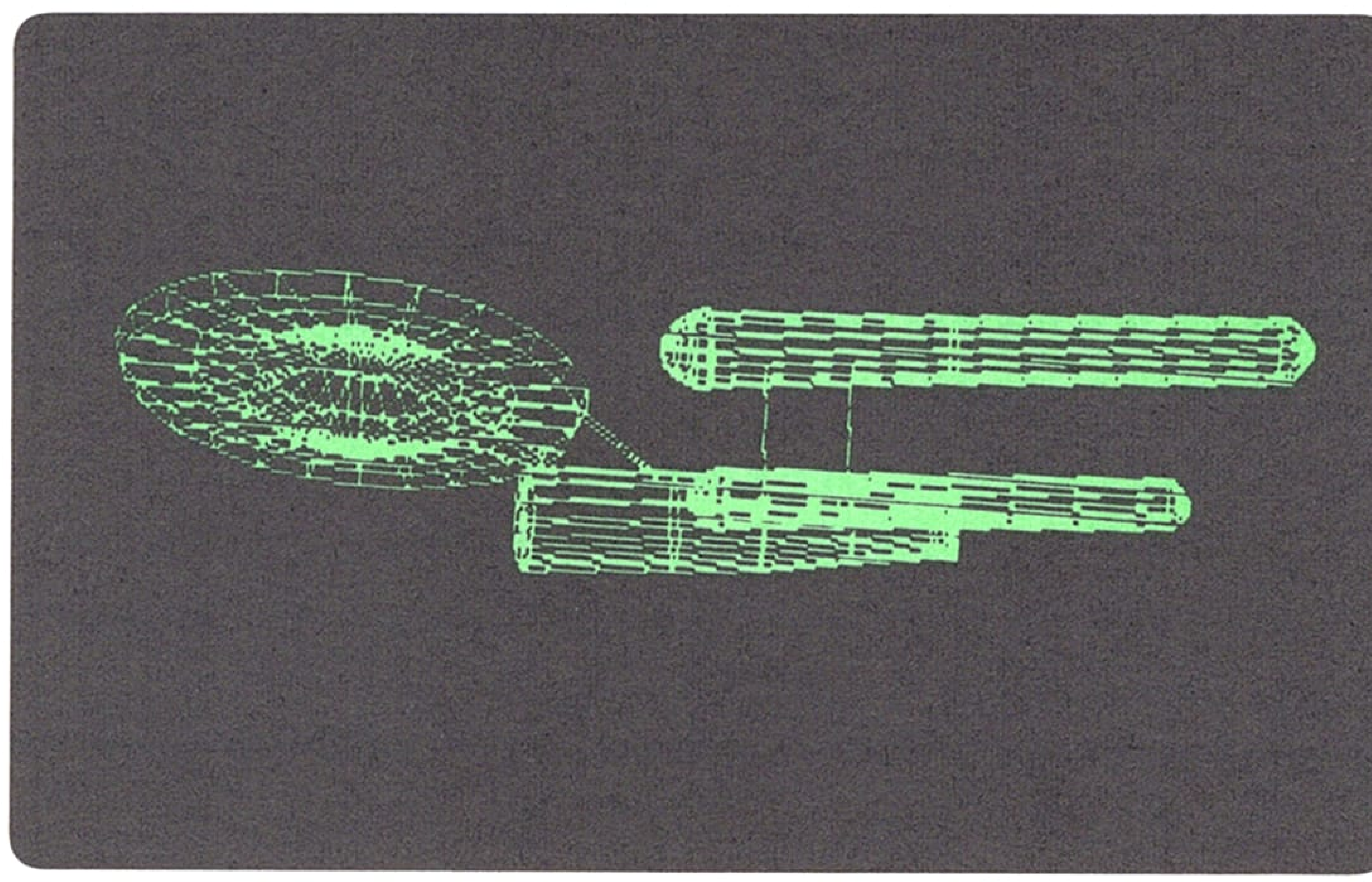
5・11 グラフィックの宇宙への出発

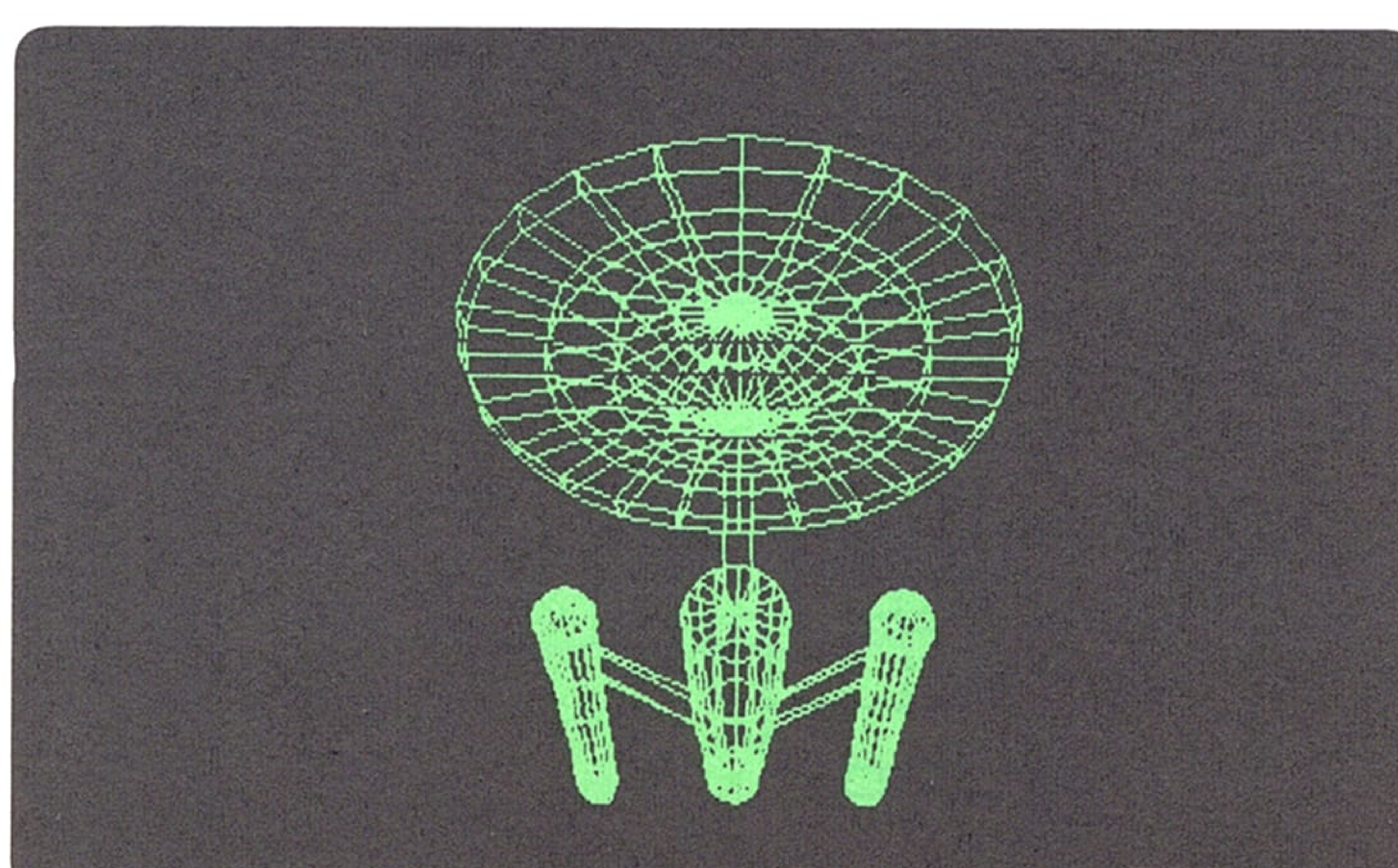
物体の３次元像をとり扱う３Ｄグラフィックスがパソコンでできるようになったことは大変な驚きです。私が大学院の学生だった十数年昔、私の専攻している分野で３Ｄグラフィックスが使われ始めました。現在はワイヤーフレーム・モデルと呼ばれているワイヤーグリッド・モデルが始めて登場して来た時は大変視覚的で衝撃を受けました。当時の大型コンピュータと私のかかわり合いは（IBM7040やIBM360）IBMカードの出し入れとプリンタ出力だけで、どこかにプロッタがついているらしいという話でしたが、どこにあったのかも良くわかりませんでした。コンピュータは非常に巨大で、建物の２層のフロアーを使っていましたが、CRTディスプレイすらなく、ワイヤーグリッド・モデルを大型コンピュータにどうのせるのか、IBMカードからだけ推測するのは大変なことでした。

この結びを書くに当たって米国の文献をひっくり返して、当時のモデルを捜し出してみました。ヘリコプターはOH-６A、ジェット機はF４であると思われますが、国家的研究のため多額の研究費用と優秀な頭脳を投入していたにもかかわらず、正直なところ、そう素晴しいものであるとは思えません。月日の流れはおそろしいものです。ＰＣ－８８０１で出力したエンタープライズ号と比較するとその理由がおわかり頂けるだろうと思います。この本を書くに当たっては常に頭の中にワイヤーフレーム・モデルがちらついていました。





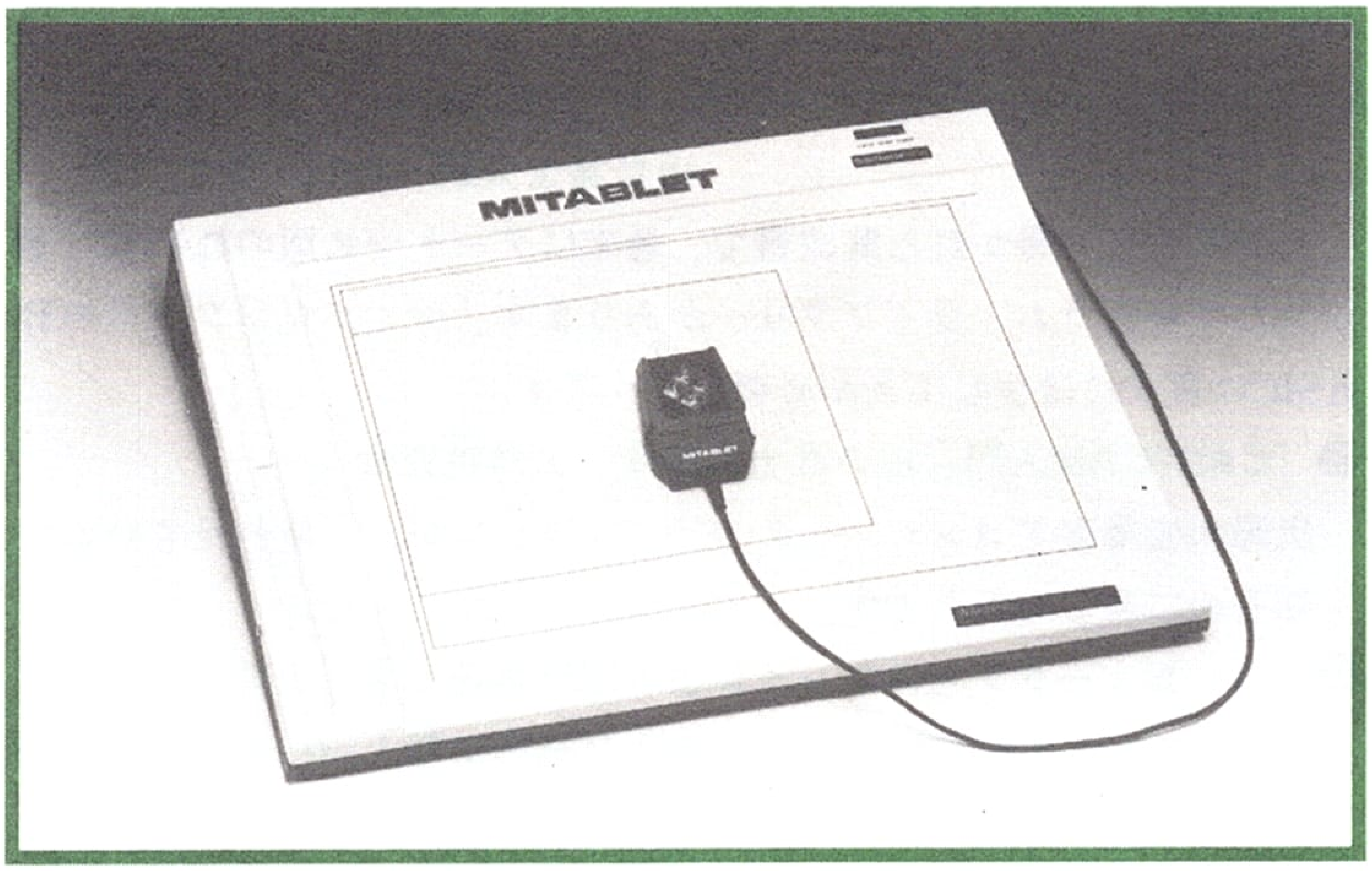




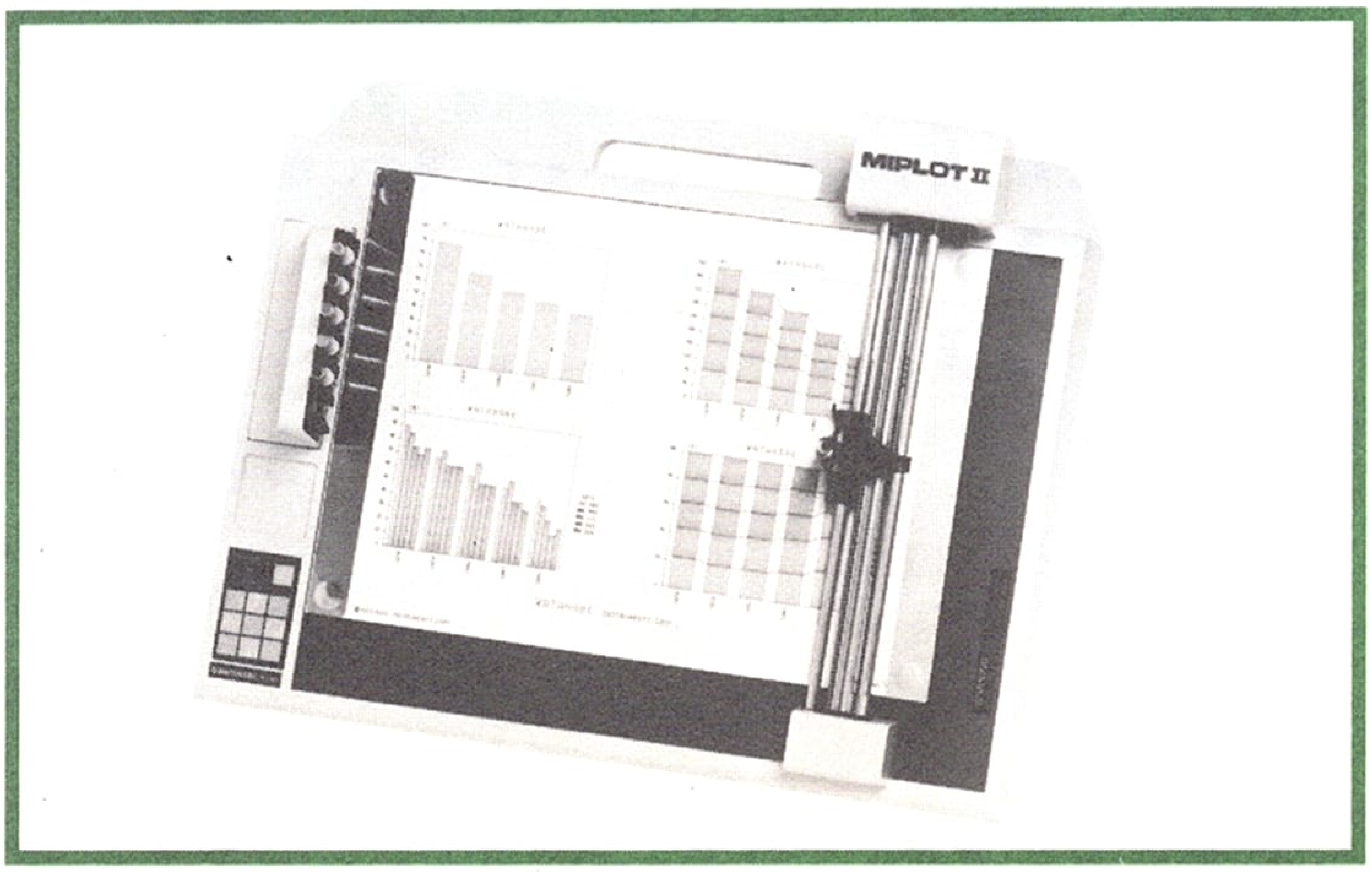
昔どうしても大型コンピュータでできなかったことをパソコンでやってみようとしたのです。ＰＣ－８８０１は私の期待以上に素晴しい能力を発揮してくれましたし、若い学生諸君の柔軟な頭脳から生まれるアイデアは、いくつもの難関を突破させてくれました。しかしバラバラのプログラムを整理し一貫した筋道をつけ読みやすいものにするのは非常に骨の折れる作業でした。今一区切りついてやっとホッとしています。

しかし、本書は入門の入門でしかありません。本当の入門はこれから始まるのです。読者の皆さんが、本書を契機にグラフィックスの広大な宇宙への旅立ちを決意されたら、著者として幸いこれに過ぐるものはありません。

最後に、本書執筆にあたっておおいに活躍してくれた渡辺測器のマイタブレット、マイプロットを紹介しておきます。



マイタブレット



マイプロット

参考文献

❶『PC-8801　グラフィックスのすべて』工藤他著　アスキー出版

大変高い水準で書かれた本で、一冊座右におかれると非常に参考になると思います。ゆき届いたプログラムが満載されています。

❷『PC-9801　ガイドブック』

校正中に出版された本なので詳しく見ている暇はありませんが、すばらしい惑星の口絵がのっています。PC-9801の解説書としてもすぐれています。

❸『MICROCOMPUTER GRAPHICS』:Roy E. Myers
Addison Wesley.

例題の図はそれほどこってはいませんが、理論の水準は高く、陰線消去などが詳しく説明されています。もともとはアップルII用に書かれました。

❹『コンピュータ・グラフィックス』D. F. Rogers、J. A. Adams
山口冨士夫訳　日刊工業新聞社

わかりやすく書かれた解説書で、巻末にダートマス版のBASICで書かれたプログラム・ライブラリーがあります。マイクロソフト系のBASICに直すには少し工夫が必要なようです。

❺『CAD/CAM入門』山口冨士夫編著　工業調査会

実際の産業界でコンピュータ・グラフィックスがどう使われているかを知るのに適した好著です。

❻『コンピュータディスプレイによる図形処理工学』山口冨士夫著
『コンピュータディスプレイによる形状処理工学 I，II』山口冨士夫著
日刊工業新聞社

上級者向きの本で、しっかりと勉強する方に向いています。

❼『コンピュータ・グラフィックス』穂坂衛著　産業図書

著者独自の理論と哲学が展開されています。非常に難しい本ですが、参考になる所が多いと思います。

補足説明

■1　座標系とウィンドウについて

もう少し先へ進むにつれて徐々に解説も加えられていますが、グラフィックスにおいては座標系の考え方が基本になりますので、ここで補足説明をしておきます。

まず、ＰＣ―8801に接続したディスプレイ上において私たちが見ることができる画面には２つあります。

テキスト画面

１つは「テキスト画面」です。これは、みなさんがプログラムを入力したりするときに使用する画面で、表示方法はキャラクタ単位になっています。

グラフィックス画面

もう１つは「グラフィックス画面」で、本書で、その寄りどころとする画面のことです。この画面については0･5節で解説がありますので、そちらを読んでいただきたいと思いますが、**テキスト画面との大きな違いは、ディスプレイへの出力形式がキャラクタ単位ではなくドット（点）単位であるということです。**そのため、ずいぶんきめ細かなカラーあるいは図形を楽しむことができます。

キャラクタ座標系
グラフィックス座標系

ところで、「テキスト画面」「グラフィックス画面」を操作するために座標系が必要になりますが、前者に導入されるものを「キャラクタ座標系」、後者に導入されるものを「グラフィックス座標系」といいます。

スクリーン座標系
ワールド座標系

この「グラフィックス座標系」はさらに「スクリーン座標系」および「ワールド座標系」という２つの座標系を持っています。

グラフィックスを勉強するためには、どうしてもこういった座標系の考え方をしっかりさせておかなければならないのですが、いわゆる**数学の座標系とＹ軸の扱いが逆である**こと、**ワールド座標系が仮想の座標系である**などの理由のため、初めての人にとっては大変とっつきにくいものになっています。

そこで本項では、簡単なテストプログラムを実際に走らせながら、座標系の考え方をイメージ的に理解していただきたいと思います。

●カラーボックスを描く

まず大切なことは、グラフィックスにおける描画は、最初に**ワールド座標に描かれ、次にそれがスクリーン座標に映し出され、私たちはこれをディスプレイ上に見ることができるということです。**

スクリーン座標系も理解に苦しむところですが、いまは、ディスプレイ上のグラフィックス座標系（たとえばカラーモードならば、横640ドット、縦200ドット）と考えておいていただいてけっこうです。

それでは、グラフィック命令を実行した場合、ワールド座標系にどのような描画がなされ、ディスプレイ画面に表示されるのか、プログラムをキーインしてテストしてみましょう。

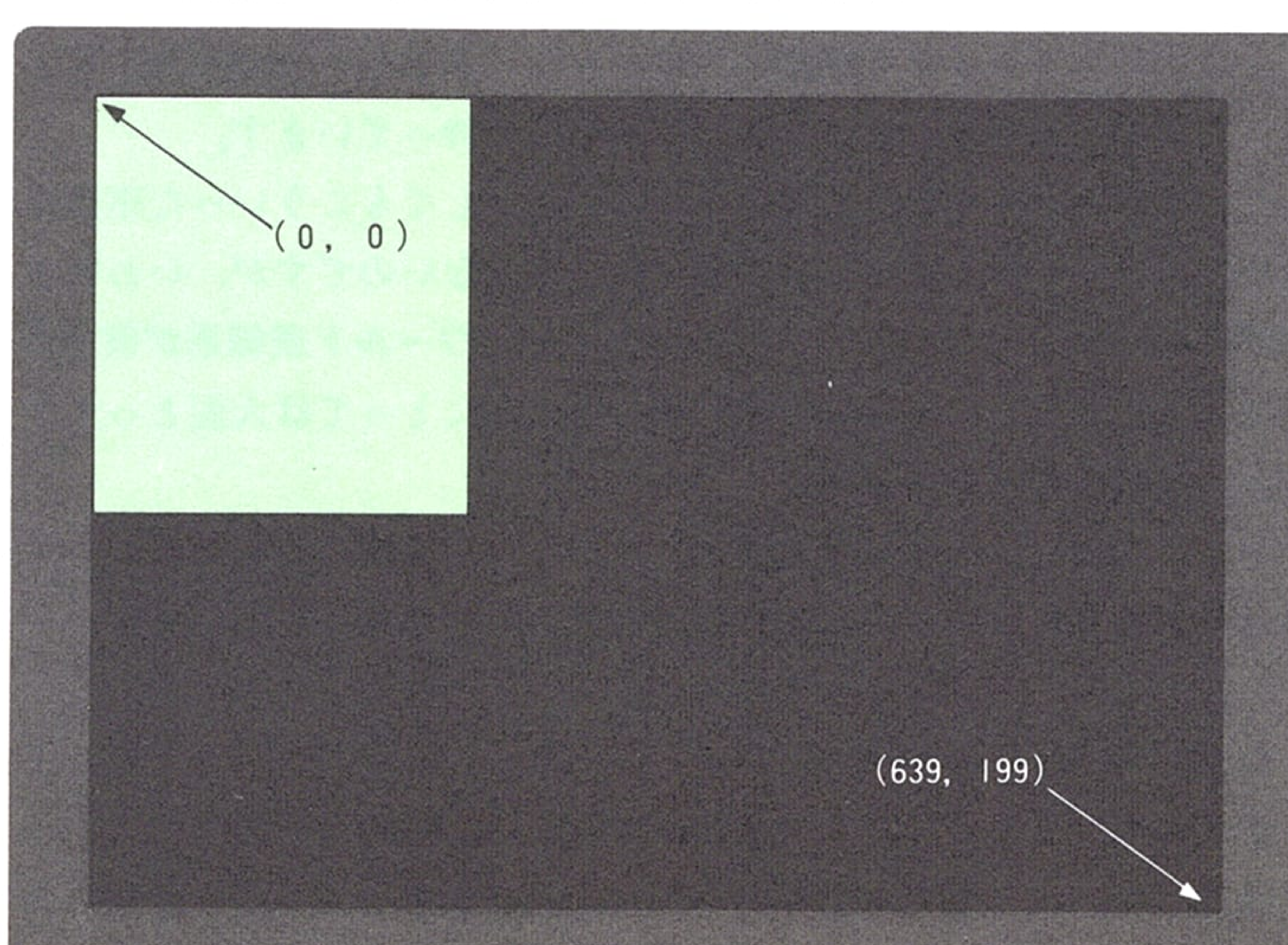
次のように入力してください。

```
100 ' sample program 01
110 ' file name ==> s01
200 '
210 GOSUB *INITIATE
220 LINE (0,0)-(200,100),6,BF
230 END
5000 '
5010 *INITIATE
5020 SCREEN 0,3 : CLS 3 : SCREEN 0,0
5030 WIDTH 80,25 : CONSOLE 0,25,0,0
5040 RETURN
```

220行のＬＩＮＥ文は、Ｎ－ＢＡＳＩＣのＬＩＮＥ文とは違いますよ。使い方は似ていますが、れっきとしたグラフィックスにおけるステートメントです（ＬＩＮＥ文の構文図は44ページにあります)。

このプログラムにいくつかルーチンを追加しますので、初期設定はサブルーチンとしておきました。

これを実行すると次のような画面になります。



ワールド座標の点(0, 0)と(200, 100)を対角線とする長方形を黄色で塗りつぶし、それがスクリーン座標系に変換され、私たちはディスプレイ上に前ページのような画面を見ることができました。

(〈注〉長方形として描かれますが、ディスプレイ画面上でのドットの比率が1：1になっていないので画面には正方形のように表示されます)

といわれても、何だか当たり前といった感じがしますね。

5020行でSCREEN文の第1パラメータは0に指定されていますからカラーモードです。ということは、ディスプレイの左上スミの座標は(0, 0)、ですから、この原点と(200, 100)を対角線とする四角形を描けば、別にワールド座標、スクリーン座標の考え方を導入しなくても当然前ページの画面のようになりそうな気がします。

しかし、この当たり前の現象の裏に隠れた秘密が非常に大切なことで、これを理解しておかないと先へ進んで混乱します。

●水色のカラーボックスを描く

ワールド座標系は、16ページの図を見てもわかるように、広大な空間を持っていて、座標指定にはマイナスも許されています。

そして、グラフィックス命令は常にこのワールド座標に働いて実行されます。

それなら負の座標を指定して、前と同じように、今度は水色のボックスを描いてみましょう。ディスプレイ上にどのように表示されるか楽しみですね。

次のプログラムを追加してキーインしてください。

```
300 '
310 GOSUB *INITIATE
320 LINE (-200,-100)-(0,0),5,BF
330 LINE (0,0)-(200,100),6,BF
340 END
```

ワールド座標系の原点(0, 0)を中心にして、第2象限に水色のカラーボックス、第4象限に黄色のカラーボックスが対称に描かれるはずです。これがスクリーン座標系にどのように反映し、ディスプレイをとおして私たちの目にどのようにうつるでしょうか。

GOTO　300 ⏎で実行してみてください。

実行結果はどうでしたか？　先ほどのプログラムを走らせたときと変わりませんね。水色のカラーボックスは表示されませんでした。

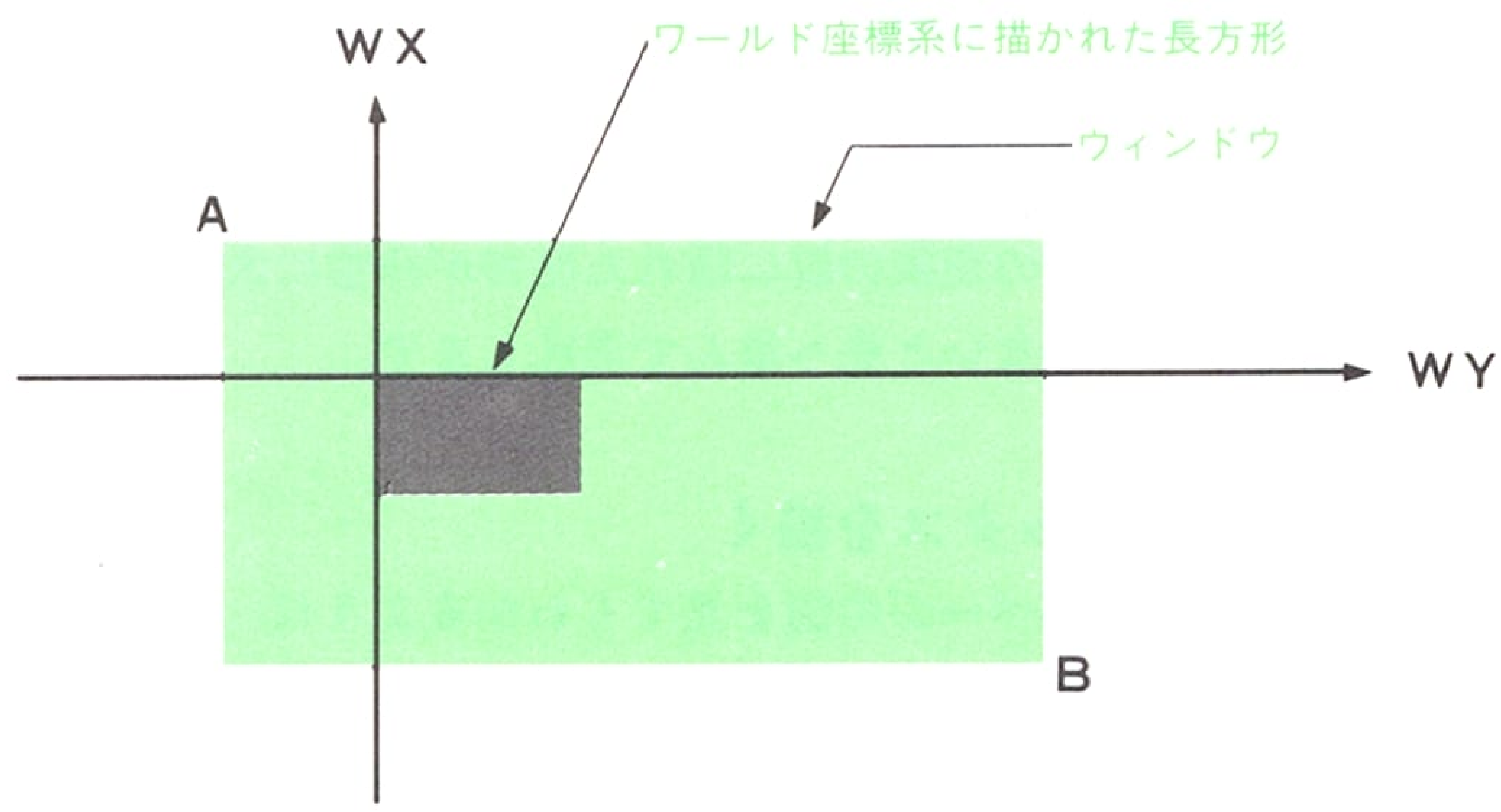
ここで登場するのがWINDOW（ウィンドウ）です。　WINDOW

ワールド座標系に描かれたグラフィックス図形をディスプレイ上に表示させるにはＷＩＮＤＯＷ文を使います。

ウィンドウとは「のぞき窓」のことですが、私たちがカメラを手にすると、広大な景色を前にして、どの範囲をフィルムにおさめようかとファインダーをのぞきますね。原理はこれと同じです。

下の図をごらんください。

これは、ワールド座標系に描かれた長方形に対して、ＡＢを対角線とするウィンドウを設定したところです。この図のウィンドウ指定ですと、長方形はディスプレイ上の真ん中、左寄りに表示されます。



ウィンドウを設定しない場合には、ウィンドウはスクリーン座標系（現在はディスプレイ座標と同じ）に一致します。

現在のディスプレイ座標は(0, 0)と(640, 200)〈注——正確には(639, 199)〉を結ぶ長方形になっていますから、これがのぞき窓になっていたのです。

つまり、ワールド座標系に描かれたグラフィックス図形を、このウィンドウで見ていたのです。ですから、ウィンドウを指定しても、これと同じ指定では先ほどの水色のカラーボックスを見ることができません。ちなみに、次のプログラムを追加して、「ＧＯＴＯ 400⏎」を実行してみてください。

```
400 '
410 GOSUB *INITIATE
420 WINDOW (0,0)-(640,200)
430 LINE (-200,-100)-(0,0),5,BF
440 LINE (0,0)-(200,100),6,BF
450 END
```

もうおわかりになると思いますが、水色のカラーボックスを見るためには、このウィンドウの範囲をもう少し広げてやらなければならないのです。

次のプログラムを入力し、やはりＧＯＴＯ文で実行してみてください。

```
500 '
510 GOSUB *INITIATE
520 WINDOW (-640,-200)-(640,200)
530 LINE (-200,-100)-(0,0),5,BF
540 LINE (0,0)-(200,100),6,BF
550 END
```

今度は画面中央に、水色と黄色の四角形が１つずつ表示されましたね。やっと念願がかないました。

これで、ワールド座標系とウィンドウの関係がご理解いただけたと思います。

ウィンドウを広範囲にとれば、ワールド座標系に描かれたグラフィックス図形はディスプレイ上では小さくなりますし、ウィンドウを狭くとれば、今度は反対にアップにうつし出されます。

次の２つのルーチンを追加してテストしてみてください。感じがつかめると思います。

```
600 '
610 GOSUB *INITIATE
620 WINDOW (-2000,-500)-(2000,500)
630 LINE (-200,-100)-(0,0),5,BF
640 LINE (0,0)-(200,100),6,BF
650 END
```

```
700 '
710 GOSUB *INITIATE
720 WINDOW (-200,-100)-(200,100)
730 LINE (-200,-100)-(0,0),5,BF
740 LINE (0,0)-(200,100),6,BF
750 END
```

なお本項ではスクリーン座標系はディスプレイ座標系とほぼ同じとして扱ってきましたが、ビューポート（35ページ）を設定すると、このビューポートがスクリーン座標系になります。

■2 ビューポートについて

グラフィックスのウィンドウと類似した考え方にビューポートという概念があります。

ウィンドウは、ワールド座標系のどの辺をトリミングして見るかということでしたが、これに対してビューポートは、グラフィックス描画をディスプレイのどの位置に表示させるかを決めるものです。

たとえば、次のプログラムを入力して実行してみてください。

```
100 ' sample program 02
110 ' file name ==> s02
200 '
210 GOSUB *INITIATE
220 WINDOW (-640,-200)-(639,199)
230 LINE (-200,-100)-(0,0),5,BF
240 LINE (0,0)-(200,100),6,BF
250 END
5000 '
5010 *INITIATE
5020 SCREEN 0,3 : CLS 3 : SCREEN 0,0
5030 WIDTH 80,25 : CONSOLE 0,25,0,0
5040 RETURN
```

実行画面を見るとすぐわかると思いますが、これは私たちが前にテストしたものと同じプログラムです。

四角形が斜めに2つ、ディスプレイの中央に表示されましたね。

前に私は、グラフィックス描画はワールド座標系に描き込まれ、それがスクリーン座標系に変換されて、それをディスプレイをとおして見る、と述べました。そして、上記のプログラムでもこれと同じ操作がなされていますが、この場合にはスクリーン座標系がディスプレイ座標系に一致していたため、ウィンドウでのぞいた図をそのままの状態で見ることになりました。

しかし、ビューポートの概念を導入すると、今度はスクリーン座標系はビューポートになりますので、ＶＩＥＷ文で指定されたディスプレイ上の位置にグラフィックス図形を見ることになります。

そして、ディスプレイ上に表示される図形の大きさはＶＩＥＷ文で指定する座標に依存し、ディスプレイ座標系をその最大値とし、それぞれ指定された位置に相似形として表示されます。

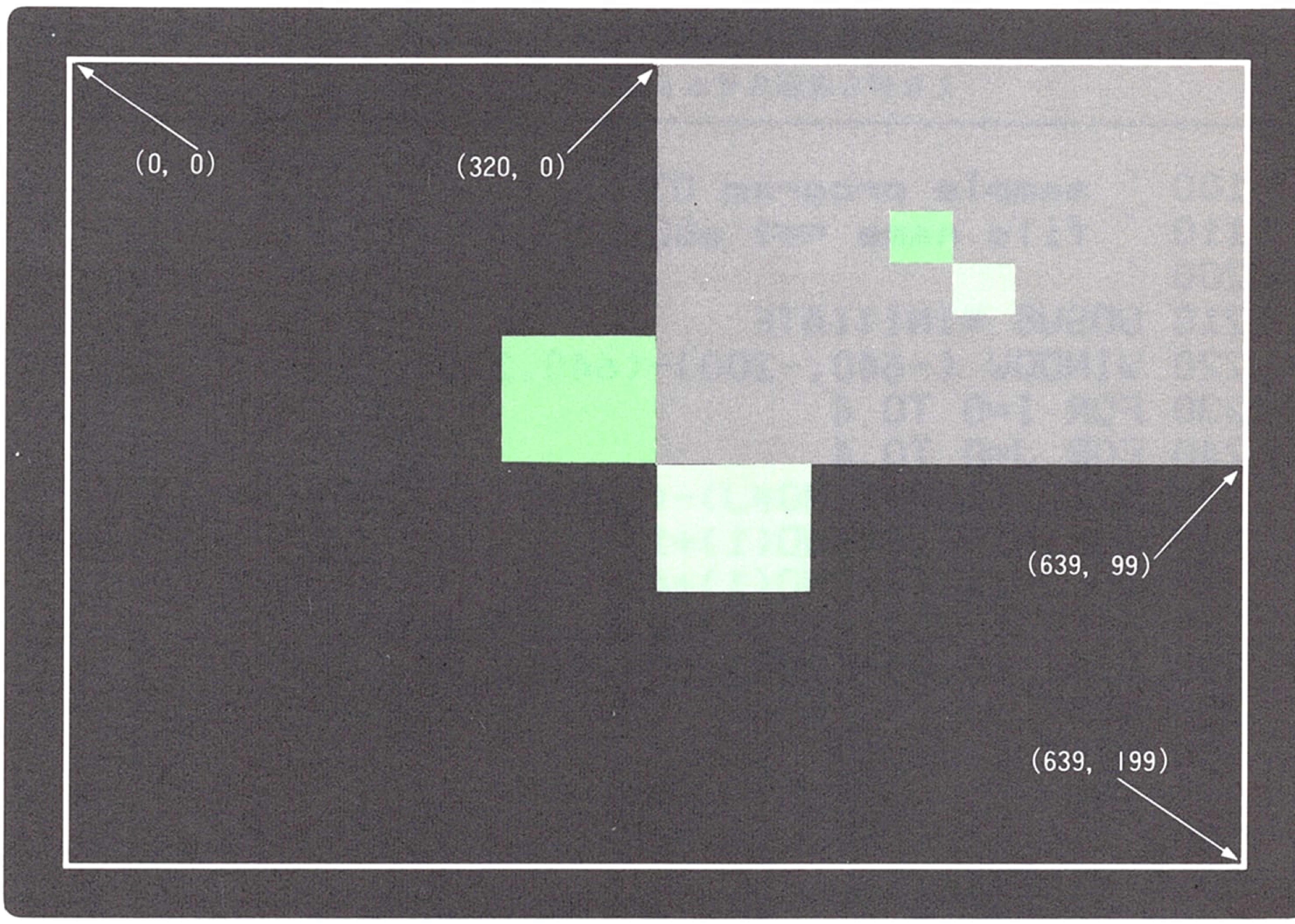
理屈をいうより実際にためしてみるほうがはやいと思いますので、

次のプログラムを追加入力して、ＧＯＴＯ文で実行してみてください。

```
300 '
310 GOSUB *INITIATE
320 WINDOW (-640,-200)-(639,199)
330 VIEW (320,0)-(639,99)
340 LINE (-200,-100)-(0,0),5,BF
350 LINE (0,0)-(200,100),6,BF
360 END
```

どうですか？　画面の中央より右斜め上の方向に、先ほどの実行結果と同じ図形が縮小されて表示されたでしょう。

次の図をごらんください。



白い太い線の枠が、ビューポートを指定しないときのスクリーン座標系です。そして、薄墨の部分がビューポート指定によるスクリーン座標系です。

このようにＶＩＥＷ文を用いてビューポートを指定することにより、ディスプレイ上の任意の位置にグラフィックス図形を表示することができます。大変便利な機能です。

ところで、ビューポート指定をせずに、上のプログラムと同じ結果を得ようとすると、ウィンドウの指定を調整しなければなりません。

次のプログラムでは、ウィンドウを変更してみました。入力して実行してみてください。だいたい同じ形が得られると思います。

```
400 '
410 GOSUB *INITIATE
420 WINDOW (-2000,-200)-(640,700)
430 LINE (-200,-100)-(0,0),5,BF
440 LINE (0,0)-(200,100),6,BF
450 END
```

しかし、トリミングが大変ですね。これに比べてＶＩＥＷ文ですと、座標がはっきりしていますから、簡単に指定できます。

次にかかげるのは、ビューポートを利用したサンプルプログラムです。私たちがいま画面に見ている図形を、ビューポートとカラーを変えながら表示させるものです。

```
100 ' sample program 03
110 ' file name ==> s03
200 '
210 GOSUB *INITIATE
220 WINDOW (-640,-200)-(640,200)
230 FOR I=0 TO 4
240 FOR J=0 TO 4
250 VIEW (100*I,30*J)-(100*(I+1),30*(J+1))
260 COL1=INT(3*RND(1)+1)
270 COL2=INT(4*RND(1)+4)
280 LINE (-200,-100)-(0,0),COL1,BF
290 LINE (0,0)-(200,100),COL2,BF
300 NEXT J
310 NEXT I
320 END
5000 '
5010 *INITIATE
5020 SCREEN 0,3 : CLS 3 : SCREEN 0,0
5030 WIDTH 80,25 : CONSOLE 0,25,0,0
5040 RETURN
```

さて、グラフィックス画面を消去する命令に「ＣＬＳ　２」があります（23ページ参照）。

いま私たちは、サンプルプログラム３を実行して25個のグラフィックス図形を画面に見ています。ＣＬＳ文のパラメータを２にするとグラフィックス画面をクリアすることになっています。ダイレクトモー

ドで次のようにキーインしてみてください。

```
cls 2
```

いかがですか？

一番右下スミの図形が消えただけですね。あとはすべてそのまま残っています。

これはなぜかといいますと、ＣＬＳ文がグラフィックス画面に対して効力を持つのは、現在アクティブなビューポートに対してだけなのです。つまり、先ほどのプログラムでは、Ｉ＝４のループでＪ＝４のとき設定されたビューポートが最終的なビューポートだったからです。

ビューポートの指定がなければ、ディスプレイ座標系がスクリーン座標系に一致し、それがビューポートになっていますから、ＣＬＳ２を実行すれば、ディスプレイ画面全体をクリアすることになります。

ですから、現在の画面をＣＬＳ文でクリアするのなら、ビューポートを次のようにディスプレイ画面全体に指定してやればよいのです。

```
view (0,0)-(639,199)
Ok
cls 2
Ok
█
```

それでは、いままでの説明を総合しますと、ウィンドウはワールド座標系に対して働き、ビューポートはディスプレイ画面に対して効力を持ち、そしてビューポートを指定するとビューポートがスクリーン座標系になり、指定がなければスクリーン座標系はディスプレイ座標に一致するということになります。さらに、私たちが見ることができるのはスクリーン座標系のグラフィックス図形だけです。

なお、ビューポートで指定する座標がディスプレイ画面を外れるようなことがありますと、イリーガル・ファンクション・コールのエラーになります。

■3 エンタープライブ号データの入力

67～68ページのエンタープライズ号のデータを入力して、グラフィックス画面に表示してみたいという方もいると思いますので、テープおよびディスクの場合について説明しておきます。

66ページの記事にも書いてありますが、テープにしてもディスクにしても、次のようにプログラムのうしろにＤＡＴＡ文をつけてください。

```
100 FOR I=1 TO 7:COLOR=(I,I):NEXT
110 SCREEN 0,3:CLS 3:SCREEN 0,0
120 CONSOLE 0,25,0,1:WIDTH 80,25
130 COLOR 7,0,0,7
140 COL=7
150 WINDOW(0,0)-(639,199):VIEW(0,0)-(639,199)
160 READ X,Y,F
170 IF F=0 THEN GOTO *FINISH
180 X=INT(X/6.72+.5):Y=299-INT(Y/7+.5)
190 IF PCNT=0 OR F<>1 THEN PSET(X,Y),COL
    :GOTO 210
200 LINE(XOLD,YOLD)-(X,Y),COL
210 XOLD=X:YOLD=Y
220 PCNT=1
230 GOTO 160
240 *FINISH
250 END
1000 DATA 425,1290,2,413,1330,1,427,1343,1
1010 DATA 459,1357,1,486,1363,1,520,1374,1
1020 DATA 569,1380,1,623,1388,1,665,1393,1
1030 DATA 715,1394,1,767,1395,1,808,1397,1
1040 DATA 851,1398,1,905,1399,1,950,1401,1
1050 DATA 976,1401,1,1014,1402,1,1058,1405,1

● この間は省略してあります

1240 DATA 786,1337,1,856,1343,1,895,1345,1
1250 DATA 929,1345,1,962,1344,1,993,1345,1
1260 DATA 1015,1345,1,1035,1345,1,1058,1345,1
1270 DATA 1085,1345,1,1112,1345,1,1148,1344,1
```

ＤＡＴＡ文は行番号1000からとするとわかりやすくてよいと思います。上のリストでは、67ページのデータの左１列を入力し終ったとこ

ろですが、このように1列終るごとに次のようにしてデータエンドである0を入力し、RUNさせてみるとよいでしょう。

```
1280 DATA 0,0,0
```

なぜ1列ごとにプログラムを走らせてみるかといいますと、かなりデータ数がありますから、全部入力し終ってからですと、正しいエンタープライズ号像が得られなかった場合、DATA文のチェックが大変だからです。

2列目は1280行から入力することになります。上のステートメントは残しておかないでください。

そして、DATA文の最後は68ページの一番右の列、次のようになってめでたくキーイン終了です。

```
3770 DATA 2230,1771,1,2250,1791,4,2244,1774,1
3780 DATA 2255,1777,1,2263,1794,1,2251,1789,1
3790 DATA 2276,1800,4,2273,1778,1,0,0,0
```

カセットテープの方は、これでRUNさせて、うまくいったらプログラムをセーブして終りです。

次に、このデータをディスクにデータファイルとして保存しておきたい方の場合ですが、ファイル作成までにどんなことがあるかわかりませんので、適当なファイル名でセーブしておいてください。

セーブできましたら、次のように不要行を削除します。

```
delete 100-250 ⏎
```

そして、シーケンシャルファイル作成のためのプログラム（行番号100－150）をキーインします。

```
100 DEFINT A-Z
110 OPEN "EPRISE.dat" FOR OUTPUT AS #1
120 READ X,Y,F
130 IF F=0 THEN CLOSE #1 : END
140 WRITE #1,X,Y,F
150 GOTO 120
1000 DATA 425,1290,2,413,1330,1,427,1343,1
1010 DATA 459,1357,1,486,1363,1,520,1374,1

        ●この間は省略してあります

3790 DATA 2276,1800,4,2273,1778,1,0,0,0
```

キーインできましたら、RUNさせてください。しばらくしてエンタープライズ号のシーケンシャルデータファイルができあがります。

次に、NEWコマンドでメモリをクリアして、次のデータ読み出しのプログラムを入力します。66ページのプログラムとは、緑色のステートメントが異なるだけです。

```
100 FOR I=1 TO 7:COLOR=(I,I):NEXT
110 SCREEN 0,3:CLS 3:SCREEN 0,0
120 CONSOLE 0,25,0,1:WIDTH 80,25
130 COLOR 7,0,0,7
140 COL=6
150 WINDOW(0,0)-(639,199):VIEW(0,0)-(639,199)
160 OPEN "EPRISE.dat" FOR INPUT AS #1
170 WHILE NOT(EOF(1))
175 INPUT #1 ,X,Y,F
180 X=INT(X/6.72+.5):Y=299-INT(Y/7+.5)
190 IF PCNT=0 OR F<>1 THEN PSET(X,Y),COL
    :GOTO 210
200 LINE(XOLD,YOLD)-(X,Y),COL
210 XOLD=X:YOLD=Y
220 PCNT=1
230 WEND
250 END
```

さあ、これで完成です。このプログラムをセーブしてから実行してみてください。

カチ、カチというディスクのアクセス音がしながら、画面にエンタープライズ号が描かれていきます。

■4　タイルストリングによる色の配合

たとえば270行のデータ文の最初の3つのデータを次のように変えたら、いったいどんな色になるか考えてみましょう。

```
270 DATA 12,34,56,00,ff,00,00,00,ff,ff,ff,00
```

まず、16進数の「１２」は青のプレーンのタイルストリング、そして「３４」は赤のプレーンの、「５６」は緑のプレーンのタイルストリングです。

これは、青（Blue）、赤（Red）、緑（Green）になっていますが、これを逆にして緑のプレーン、赤のプレーン、青のプレーンの順に並べてみます。

緑 56	赤 34	青 12	3ビット表現になおすと	10進数になおすと	このカラーコードに対応する色は
0	0	0	000	0	黒
1	0	0	100	4	緑
0	1	0	010	2	赤
1	1	1	111	7	白
0	0	0	000	0	黒
1	1	0	110	6	黄
1	0	1	101	5	水色
0	0	0	000	0	黒

上位ビット ↓
下位ビット

このように、3原色を8ビット表現になおし、さらに3枚のプレーンの各ビットを対応させて3ビット表現にし、これを10進数に変換して22ページのカラーコード表を見ると、右端の色が得られます。

つまり、このタイルストリングでは、

黒　緑　赤　白　黒　黄　水色　黒　黒　緑　赤　……

といった配分の円を見ることができます。

黒が続きますので、黒い縞がタテに何本か走り、その間に奇妙な色の帯が展開します。

みなさん、実行してためしてみてください。

■5　タイルパターンのテストプログラム

プログラム3-8で作ったタイルパターンが、いったいどんな色になるのか、このプログラムに113ページのプログラム3-6をマージしてテストしてみましょう。手順はディスク内蔵タイプの場合を例にとって説明します。

まず、プログラム3-6をロードしてください。

次に、不要行を削除します。

```
delete 260-280
```

次のREM文を入力してください。

```
90 '
```

このプログラムをプログラム3-8にマージしますので、リナンバリ

ングをします。

```
renum 700,90
```

リナンバリングできたら、これをアスキーセーブしておきます。

```
save "prg3-6",a
```

さあ、次は、プログラム3-8をロードします。

ロードできましたら、これにプログラム3-6をつなげます。

```
merge "prg3-6"
```

これで準備ができました。

それではＲＵＮしてください。「ＹＭＡＸ？」には「７」を入力して、119ページの方法でタイルパターンを作ってください。

最初のカラーコードを入力すると、画面の右下に７行のＤＡＴＡが表示されます。これはテキスト画面への出力ですから、スクリーンエディタでＤＡＴＡ文にすることができます。最初の１行目だけを例にとって説明しますから、残りの６行についてはみなさんが同じ要領で行なってください。

「ＤＡＴＡ」という表示の前へカーソルを移動し１、０、０、０とキーインします。

```
1000■     DATA  AB  66  1E
```

↑ このアキは何桁でもかまいません

さらにカーソルを右へ移動し、データ間にコンマをキーインします。

```
1000      DATA  AB, 66, 1E
```

上の画面のような状態になったら、リターンキーを押してください。これで、行番号1000のＤＡＴＡ文が出来上がりました。

同様に、1010～1060行を完成させてください。

ＤＡＴＡ文ができましたら、ダイレクトモードで次のステートメントを実行します。

```
go to 700
```

どうですか？　あなたの作ったタイルパターンが、７つの円の中に表示されましたね。このような方法でタイルパターンの配色をテストできますので、いろいろためしてみてください。

□6　上部構造物のデータファイル

ここでは、126ページから130ページまでのデータをシーケンシャルファイルとして保存しておく方法を説明しますが、その前に、124ページのプログラム3-12で、ステートメントを変更するともう少し見やすくなる個所がありますので、補足しておきます。

●ウィンドウとタイルストリングの変更

150行にＷＩＮＤＯＷ文がありますが、このウィンドウですと車体部分が画面の下のほうに表示されてしまいます。

ウィンドウを次のように変更すると、画面中央に表示され大変見やすくなります。

```
150 WINDOW(-100,0)-(700,500):VIEW(0,0)-(639,199)
```

次に、340行のタイルストリングですが、ＣＨＲ＄関数のカッコ内の16進数に「＆Ｈ」がついていないため、たとえば「ＡＡ」は単純変数として扱われますので、この値は０となりますから、ＣＨＲ＄（ＡＡ）は結局ＣＨＲ＄（０）と等価になります。

したがって、340行のカッコ内の値でカラー効果に影響するのは、10進数の「15」だけとなります。これは16進数では「Ｆ」ですから、緑のプレーンのタイルストリングがＣＨＲ＄（＆Ｈ０Ｆ）といいかえることができます。

一方、他の色のプレーンはＣＨＲ＄（０）となっていますので、全体として、緑および黒でペイントされますので、見た感じは125ページの画面のようになります。

もし、340行の「ＡＡ」「ＦＦ」を生かしたいという方は、次のように変更してテストしてみてください。

```
340 PAINT (X,Y),CHR$(&HAA)+CHR$(0)+CHR$(&H15)
    +CHR$(&HFF)+CHR$(0)+CHR$(&HAA),7
```

この変更を加えてプログラムを実行しますと、水色の車体部分を見ることができます。

●データファイルをつくる

プログラム3-12の100―360行のステートメントのうしろに、126ページから130ページまでのＤＡＴＡ文をキーインできましたか？

入力が終りましたら、とりあえずこのプログラムをセーブしておいてください。

セーブできたら、ダイレクトモードで次のように入力し、不要行を削除します。

```
delete 100-360
```

そして、次のプログラムを入力してください。

```
100 DEFINT A-Z
110 RESTORE *DAT1
120 OPEN "TANK1.dat" FOR OUTPUT AS #1
130 READ X,Y,F
140 IF X=0 THEN GOTO 170
150 WRITE #1,X,Y,F
160 GOTO 130
170 CLOSE #1
200 '
210 RESTORE *DAT2
220 OPEN "TANK2.dat" FOR OUTPUT AS #1
230 READ X,Y,F
240 IF X=0 THEN GOTO 270
250 WRITE #1,X,Y,F
260 GOTO 230
270 CLOSE #1
300 '
310 RESTORE *DAT3
320 OPEN "TANK3.dat" FOR OUTPUT AS #1
330 READ X,Y
340 IF X=0 THEN GOTO 370
350 WRITE #1,X,Y
360 GOTO 330
370 CLOSE #1
380 END
```

緑色のところが、それぞれ「DATA 1」「DATA 2」「DATA 3」に対するファイル名です。

380行よりうしろには、あなたが入力したDATA文が続いているはずです。

「DATA 1」「DATA 2」「DATA 3」のDATA文の一番最後は123ページの説明にあるように必ずゼロを並べてください。

間違いがないようでしたらRUNしてください。3つのデータファイルが出来上がります。

●ファイルからデータを読み出す

さあ、それではデータファイルを使って車体部分および上部構造物をグラフィックス画面に描いてみようと思いますが、その前に、前ページのデータファイル作成プログラムの必要な方は、これをセーブしてから、メモリをクリアして次のプログラムを入力してください。

```
100 ' file name ==> tank1.bas
110 ' save "tank1.bas",a
120 '
130 FOR I=1 TO 7:COLOR=(I,I):NEXT I
140 SCREEN 0,3:CLS 3:SCREEN 0,0
150 WIDTH 80,25:CONSOLE 0,25,0,1
160 COLOR 7,0,0,7
170 COL=7
180 WINDOW(-100,0)-(700,550):VIEW(0,0)-(639,199)
190 '
200 OPEN "TANK1.dat" FOR INPUT AS #1
210 GOSUB *DRAW
220 OPEN "TANK2.dat" FOR INPUT AS #1
230 GOSUB *DRAW
240 OPEN "TANK3.dat" FOR INPUT AS #1
250 GOSUB *PCL
260 END
270 '
280 *DRAW
290 WHILE NOT(EOF(1))
300 INPUT #1,X,Y,F
310 IF PCNT=0 OR F<>1 THEN PSET(X,Y),COL
    :GOTO 330
320 LINE (XOLD,YOLD)-(X,Y),COL
330 XOLD=X:YOLD=Y
340 PCNT=1
350 WEND
360 PCNT=0:CLOSE #1
370 RETURN
380 '
390 *PCL
400 WHILE NOT(EOF(1))
410 INPUT #1,X,Y
420 PAINT (X,Y),CHR$(&HAA)+CHR$(0)+CHR$(&H15)
    +CHR$(&HFF)+CHR$(0)+CHR$(&HAA),COL
430 WEND
440 RETURN
```

初期設定はプログラム3－12と同じですし、また描画手順もほとんど同じです。

データファイルをＯＰＥＮしてからサブルーチンへ飛び、ＷＨＩＬＥ～ＷＥＮＤ文を使って、データ末尾まで描画が続けられます。

入力できたらテストしてみてください。125ページと同じ図を得ることができます。

7　可動部分を画面中央へ

プログラム4－1を実行してみるとわかると思いますが、可動部分が、ディスプレイの左下のほうに小さく表示されます。

これはなぜかといいますと、160行から200行までのＳＣＲＥＥＮ文を実行しますと、そのつどウインドウおよびビューポートが初期化されてしまうためです。

つまり、ビューポートが常に（０，０）－（639,199）で変化しないため、サブルーチンでの描画が、ワールド座標系に描いていながらも、ディスプレイ座標系に描いたのとなんら変わりなくなってしまったのです。

これを防ぐには、ウィンドウをそのつど設定しなおしてやる必要があります。

ウィンドウ設定をサブルーチンにし、150行から210行を次のように変更します。

```
150 '
155 SCREEN 1,0,0,1:GOSUB *VPORT
160 RESTORE 6000:X0= 20:GOSUB *GDRAW
165 SCREEN 1,0,0,1:GOSUB *VPORT
170 RESTORE 7000:X0=-20:GOSUB *GDRAW
175 SCREEN 1,0,1,2:GOSUB *VPORT
180 RESTORE 6000:X0=  0:GOSUB *GDRAW
185 SCREEN 1,0,1,2:GOSUB *VPORT
190 RESTORE 7000:X0=  0:GOSUB *GDRAW
195 SCREEN 1,0,2,4:GOSUB *VPORT
200 RESTORE 6000:X0=-20:GOSUB *GDRAW
205 SCREEN 1,0,2,4:GOSUB *VPORT
210 RESTORE 7000:X0= 20:GOSUB *GDRAW
```

入力できましたら、次に示すサブルーチンをキーインしてください。

```
8000 '
8010 *VPORT
8020 WINDOW(-50,120)-(150,200)
8030 RETURN
```

これでプログラムを実行しますと、画面中央に可動部分が適度の大きさに表示されます。

みなさんも、8020行のウィンドウをいろいろ変えてテストしてみてください。

なお、120行のステートメントは、あってもなくてもかまいません。

■8 ディスク用プログラムリスト

次は、プログラム4－2のディスク版です。

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WINDOW (-1280,-800)-(1280,800)
120 VIEW (0,0)-(639,199)
130 CONSOLE 0,25,0,1:WIDTH 80,25
140 COLOR 7,0,0,7
150 COL=7
160 '
170 FOR K=1 TO 20
180 RAD=-3.14159*K*36/180
190 OPEN "TANK1.dat" FOR INPUT AS #1
200 GOSUB *GDRAW
210 OPEN "TANK2.dat" FOR INPUT AS #1
220 GOSUB *GDRAW
230 NEXT K
240 END
250 '
260 *GDRAW
270 WHILE NOT(EOF(1))
280 INPUT #1,X,Y,F
290 W=X*COS(RAD)+Y*SIN(RAD)
300 Y=-X*SIN(RAD)+Y*COS(RAD)
310 X=W
320 IF F>1 THEN PSET(X,Y),COL::GOTO 340
330 LINE (XOLD,YOLD)-(X,Y),COL
340 XOLD=X:YOLD=Y
350 WEND
360 CLOSE #1:RETURN
```

□9　ディスク用プログラムリスト

次のリストは、143ページのプログラム４－３をディスク用に変更したものです。

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WINDOW (-50,100)-(650,450)
120 VIEW (0,0)-(639,199)
130 CONSOLE 0,25,0,1:WIDTH 80,25
140 COLOR 7,0,0,7
150 COL=7
160 '
170 GOSUB *GDRAW1
180 FOR K=1 TO 20
190 GOSUB *GDRAW2
200 NEXT K
210 END
220 '
230 *GDRAW1
240 OPEN "TANK1.dat" FOR INPUT AS #1
250 WHILE NOT(EOF(1))
260 INPUT #1,X,Y,F
270 IF F>1 THEN PSET(X,Y),COL:GOTO 290
280 LINE (XOLD,YOLD)-(X,Y),COL
290 XOLD=X:YOLD=Y
300 WEND
310 CLOSE #1:RETURN
320 '
330 *GDRAW2
340 OPEN "TANK2.dat" FOR INPUT AS #1
350 WHILE NOT(EOF(1))
360 INPUT #1,X,Y,F
370 RAD=3.14159*K*18/180
380 W=(X-223)*COS(RAD)+(Y-259)*SIN(RAD)
390 Y=-(X-223)*SIN(RAD)+(Y-259)*COS(RAD)
400 X=W
410 X=X+223:Y=Y+259
420 IF F>1 THEN PSET(X,Y),COL:GOTO 440
430 LINE(XOLD,YOLD)-(X,Y),COL
440 XOLD=X:YOLD=Y
450 WEND
460 CLOSE #1:RETURN
```

◘10 ディスク用プログラムリスト

次は、148ページのプログラム４－５のディスク使用の方のためのリストです。

```
100 SCREEN 0,3:CLS 3:SCREEN 0,0
110 WINDOW (0,0)-(639,399)
120 VIEW (0,0)-(639,199)
130 CONSOLE 0,25,0,1:WIDTH 80,25
140 COLOR 7,0,0,7
150 COL=7
160 '
170 FOR K=3 TO 7
180 OPEN "TANK1.dat" FOR INPUT AS #1
190 GOSUB *GDRAW
200 OPEN "TANK2.dat" FOR INPUT AS #1
210 GOSUB *GDRAW
220 NEXT K
230 END
240 '
250 *GDRAW
260 WHILE NOT(EOF(1))
270 INPUT #1,X,Y,F
280 X=X/K:Y=Y/K+230-60*(K-3)
290 IF F>1 THEN PSET(X,Y),COL:GOTO 310
300 LINE (XOLD,YOLD)-(X,Y),COL
310 XOLD=X:YOLD=Y
320 WEND
330 CLOSE #1:RETURN
```

著者略歴

脇　英世（わき　ひでよ）
1947年東京生まれ。早稲田大学理工学部電子通信学科卒業。同大学大学院博士課程修了。工学博士。
現在，東京電機大学工学部電気通信工学科助教授。
著書に
『マイコン標準インターフェース』オーム社。『ビジネスマンのための実践ベーシック教室』東洋経済新報社，『ＦＭ８操縦法』ラジオ技術社，『マイコンによる知的生産の技術』講談社，『日本語ワード・プロセッサー入門』講談社。

ＰＣ－８８０１mkII グラフィックス入門

著　者　**脇　英世**

発行者　**田村正隆**

発行所　**株式会社　ナツメ社**
東京都千代田区神田神保町1－52(〒101)
電話　03(291)1257（代表）
振替　東京３－58661
印　刷　**ラン印刷社**
製　本　**三修紙工**

ISBN4-8163-0361-8 Printed in Japan

PC-8801mkII
グラフィックス入門

発行――1984年3月1日
著者――脇英世
発行者――田村正隆
発行所――株式会社ナツメ社
郵便番号101
東京都千代田区神田神保町1-52
電話〈03〉291-1257
振替 東京3-58661
〈落丁・乱丁本はお取り替えします〉

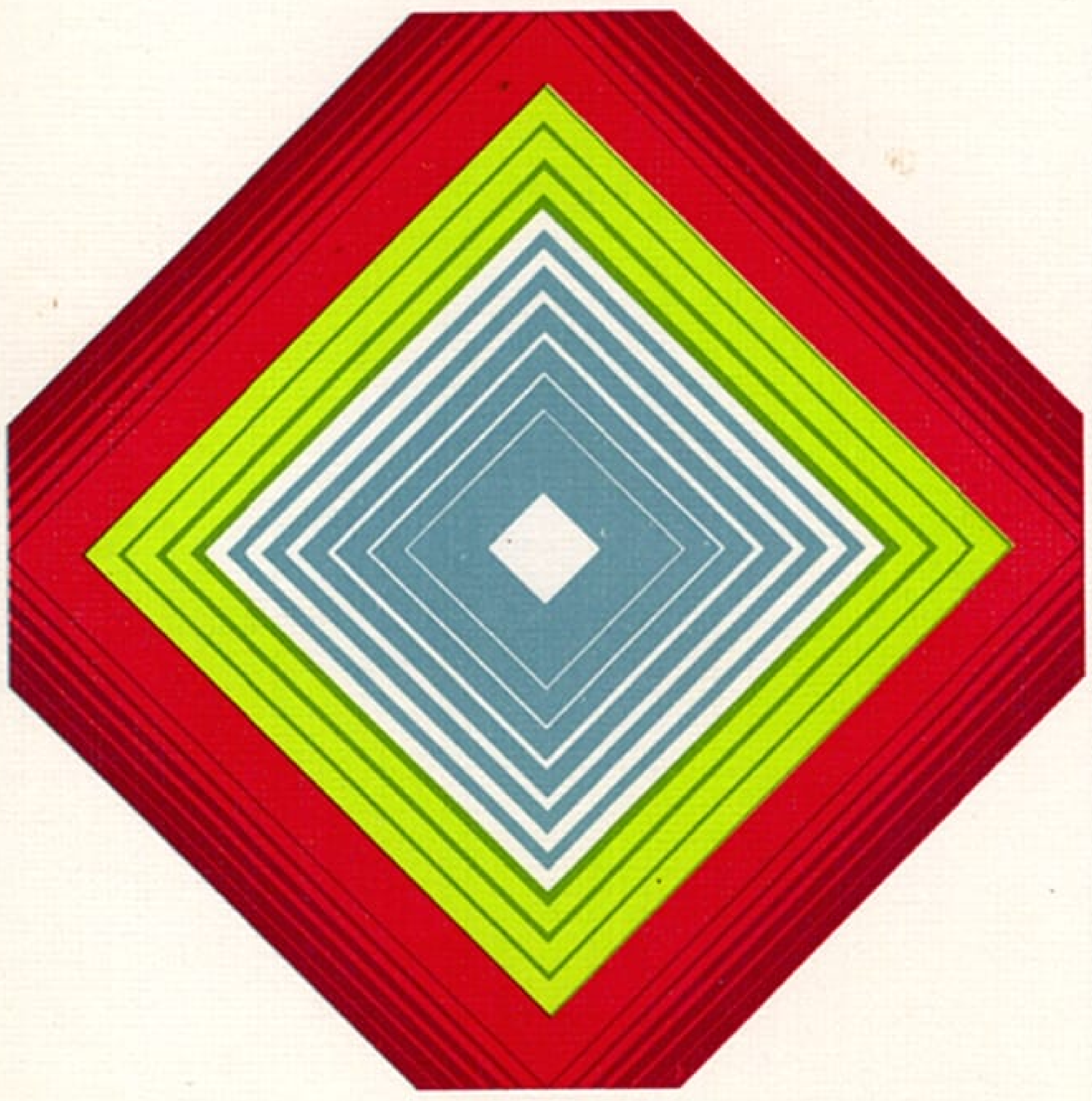
定価――1600円

PC-8801mkII

グラフィックス入門

［目次より］

ナツメ社
定価＝1600円



ISBN4-8163-0361-8 C2054 ¥1600E